(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 1 128 600 A1

(12) EUROPÄISCHE PATENTANMELDUNG

(43) Veröffentlichungstag:
29.08.2001 Patentblatt 2001/35

(51) Int Cl.7: **H04L 12/24**, H04L 29/06, H04Q 11/04

(21) Anmeldenummer: **00103798.5**

(22) Anmeldetag: **23.02.2000**

(84) Benannte Vertragsstaaten:
AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE
Benannte Erstreckungsstaaten:
AL LT LV MK RO SI

(71) Anmelder: **Tektronix, Inc.**
Beaverton, OR 97077-0001 (US)

(72) Erfinder:
- **Erhardt, Jörg**
  12109 Berlin (DE)
- **Kittan, Jens**
  16737 Schwante (DE)
- **Borgert, Wolfgang**
  10961 Berlin (DE)

(74) Vertreter: **Schurack, Eduard F. et al**
Hofstetter, Schurack & Skora
Balanstrasse 57
81541 München (DE)

(54) Verfahren zum Erstellen eines Kommunikationsablaufs zwischen mindestens zwei Instanzen und Protokolltester hierfür

(57) Das vorliegende Verfahren ist gekennzeichnet durch folgende, am Protokolltester ausführbare Schritte: a) Auswählen der an der Kommunikation beteiligten Instanzen; b) Auswählen einer Protokollschicht, auf deren Grundlage die Kommunikation zwischen den ausgewählten Instanzen ablaufen soll; c) Auswählen derjenigen abstrakten Kommunikationsschnittstellen der Protokollschicht, die an der Kommunikation beteiligt sind; d) Auswählen der Kommunikationsdaten; e) automatisches Erstellen eines zwischen den mindestens zwei Instanzen ausführbaren Kommunikationsablaufs durch den Protokolltester, auf der Grundlage des Auswahlen in den Schritten a) bis d), wobei die Auswahl von Schritt c) und/oder Schritt d) graphisch erfolgt und den dabei auswählbaren Parametern Beschreibungsdateien zugeordnet sind, die in Schritt e) zur Erstellung eines zwischen den Instanzen ausführbaren Kommunikationsablaufs verwendet werden. Die Erfindung betrifft weiterhin einen Protokolltester, in dem das erfindungsgemäße Verfahren realisiert ist.

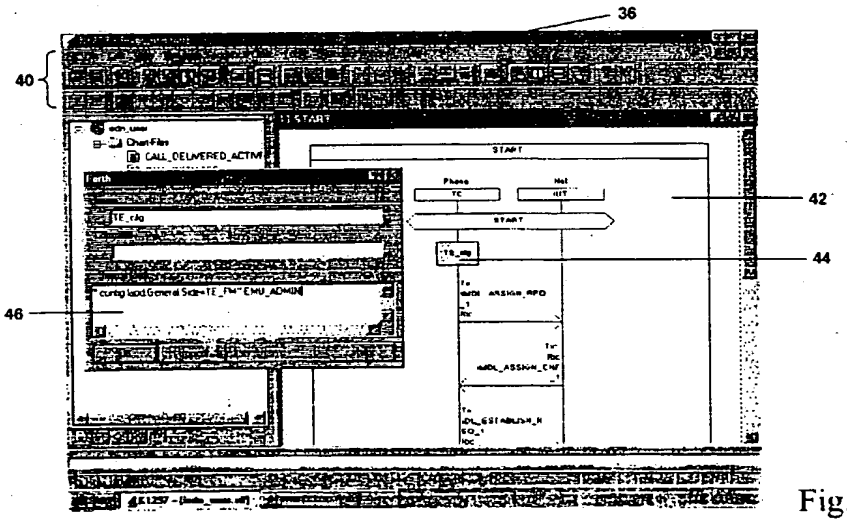


Fig. 6

EP 1 128 600 A1

Beschreibung

[0001] Die vorliegende Erfindung betrifft ein Verfahren zum Erstellen eines Ablaufs einer zwischen mindestens zwei Instanzen ablaufenden Kommunikation, wobei eine Instanz ein Protokolltester ist. Sie betrifft überdies einen Protokolltester, in dem dieses Verfahren realisiert ist.

[0002] Im Bereich des Protokolltests ist es nötig, den Kommunikationsablauf, durch den ein Test beschrieben wird, eindeutig zu spezifizieren, so daß dieser Ablauf automatisch maschinell ausgeführt werden kann. Sprachen, wie zum Beispiel TTCN (Tree and Tabular Combined Notation) ermöglichen dies, sind aber komplex und für einen ungeübten Leser schwer zu verstehen. TTCN hat sich im Bereich des Conformance Testing durchgesetzt, da diese Tests sehr umfangreich sind und TTCN solche umfangreichen Tests gut unterstützt. Daneben gibt es diverse proprietäre Testbeschreibungssprachen. Um die Verständlichkeit zu erleichtern, wird zu Dokumentations- und Beschreibungszwecken einfacher Abläufe die standardisierte Sprache MSC (Message Sequence Charts) verwendet. Detailliertere Angaben zu MSC können der ITU-T Z.120 entnommen werden, die durch diese Bezugnahme in den Offenbarungsgehalt der vorliegenden Anmeldung aufgenommen wird. Bei MSC handelt es sich um genormte Ablaufdiagramme, auch Pfeildiagramme oder X-Diagramme genannt. Sie sind unabhängig von Programmierkenntnissen zu verstehen. Eine automatische Ausführung von mit MSC beschriebenen Kommunikationen ist auf Protokolltestern jedoch nicht möglich. Um ausführbare Tests zu erhalten müssen daher sogenannte Scripts geschrieben werden, was eine Einarbeitung des Anwenders in die jeweilige Programmiersprache erforderlich macht. Eine allgemein verständliche Dokumentation muß zusätzlich erstellt werden. Für einen Test müssen daher getrennt zum einen eine graphische und textuelle Dokumentation erstellt werden, zum anderen ein Source-Code oder ein ausführbarer Binary.

[0003] Aus diesem Stand der Technik ergeben sich eine Vielzahl von Nachteilen: Häufig ist eine Konvertierung bestehender Tests nötig, so daß die Gefahr von Inkonsistenzen besteht. Konfigurationsinformationen sind oft in den Spezifikationen der Testkommunikationen nicht oder zumindest nicht maschinenlesbar oder nicht menschenlesbar enthalten. Die verwendeten Sprachen stellen häufig proprietäre Ansätze dar, die von Gerät zu Gerät verschieden sind und neu erlernt werden müssen. Der Anwender wird nicht oder nur rudimentär bei der Erstellung der Nachrichten und Ereignisse mit Protokollwissen unterstützt.

[0004] Der vorliegenden Erfindung liegt daher die Aufgabe zugrunde, die Nachteile der oben dargestellten, aus dem Stand der Technik bekannten Vorgehensweise zu überwinden.

[0005] Diese Aufgabe wird gelöst durch ein Verfahren gemäß Anspruch 1.

[0006] Gemäß einem weiteren Aspekt der Erfindung wird ein Protokolltester zur Verfügung gestellt mit den Merkmalen von Anspruch 8.

[0007] Die Erfindung erlaubt es, den gewünschten Kommunikationsablauf in einer leicht verständlichen graphischen, standardisierten Form, beispielsweise MSC, zu erstellen. Durch die erfindungsgemäße Lösung ist es nunmehr möglich, die Kommunikationsschnittstellen und/oder die Kommunikationsdaten graphisch zu spezifizieren. Mit MSC selbst allein ist dies nicht möglich. Dadurch daß den graphisch auswählbaren Parametern Beschreibungsdateien zugeordnet sind, wird die Möglichkeit bereitgestellt, daß der Protokolltester automatisch die vom Benutzer ausgewählten Parameter in eine ausführbare Version eines Kommunikationsablaufs transformiert. Durch die graphische Bereitstellung von Parameterzusammenstellungen, aus denen der Benutzer auswählen kann, wird der Benutzer in jeder Phase mit konfigurationsspezifischer und protokollspezifischer Information unterstützt. Neben einer kurzen Einweisung in einen Graphikstandard, beispielsweise MSC, benötigt der Benutzer kein weiteres Vorwissen, um das erfindungsgemäße Verfahren erfolgreich anwenden zu können. Die graphische Darstellung erfolgt auf einer Anzeigeeinheit, beispielsweise einem Monitor oder einem Bildschirm.

[0008] Dokumentation und Implementierung wird auf diese Weise in einem Vorgang erledigt, wobei der Benutzer immer die Dokumentationssicht auf einer graphischen Benutzeroberfläche sieht.

[0009] Die Bedienungsfreundlichkeit für den Benutzer steigt umso mehr, je mehr Auswahlen dem Benutzer graphisch zur Verfügung gestellt werden. In einer besonders bevorzugten Ausführungsform der Erfindung sind daher, mit Bezug auf das erfindungsgemäße Verfahren, alle Auswahlschritte gemäß a) bis d) graphisch unterstützt, wobei allen, mit den Auswahlmitteln auswählbaren Parametern Beschreibungsdateien zugeordnet sind, die dann in Schritt e) zur Erstellung eines zwischen den Instanzen ausführbaren Kommunikationsablaufs verwendbar sind.

[0010] Die abstrakten Kommunikationsschnittstellen umfassen vorzugsweise sogenannte Service Access Points (SAPs), die Kommunikationsdaten vorzugsweise sogenannte Protocol Data Units (PDUs) und/ oder sogenannte Abstract Service Primitives (ASPs). Hierbei handelt es sich um Primitive, d. h. Datenpakete, mit denen sich verschiedene Kommunikationsschichten derselben Instanz untereinander verständigen. Ein SAP ist demnach ein Punkt, an dem verschiedene ASPs ausgetauscht werden können. Vorzugsweise enthalten ASPs PDUs, wobei üblicherweise jede PDU einzeln zu erstellen ist.

[0011] Die Auswahl der Kommunikationsdaten kann zwei Teilschritte umfassen, zunächst das graphische Auswählen eines Datenformats, dann den graphischen Aufbau einer Kommunikationsabfolge zwischen den beteiligten Instanzen.

[0012] Hinsichtlich des zuletzt genannten Teilschritts kann die Möglichkeit vorgesehen werden, Source-Code einzu-

geben.

[0013] Die zuvor erwähnten Merkmale der Erfindung gelten in entsprechender Weise für einen erfindungsgemäßen Protokolltester.

[0014] Weitere vorteilhafte Weiterbildungen der Erfindung sind in den Unteransprüchen definiert.

[0015] Ein Ausführungsbeispiel wird im folgenden, unter Hinweis auf die beigefügten Zeichnungen, näher beschrieben. Es stellen dar:

Figur 1 eine erste graphische Benutzeroberfläche, wie sie bei dem erfindungsgemäßen Verfahren Anwendung findet;

Figur 2 eine zweite graphische Benutzeroberfläche, wie sie bei dem erfindungsgemäßen Verfahren Anwendung findet;

Figur 3 die Benutzeroberfläche von Figur 2 in einem anderen Darstellungsmodus;

Figur 4 die Benutzeroberfläche von Figur 2 in einem weiteren Darstellungsmodus;

Figur 5 eine dritte graphische Benutzeroberfläche, wie sie bei dem erfindungsgemäßen Verfahren Anwendung findet;

Figur 6 die Benutzeroberfläche von Figur 5 in einem anderen Darstellungsmodus; und

Figur 7 die Benutzeroberfläche von Figur 5 in einem weiteren Darstellungsmodus.

[0016] Figur 1 zeigt eine graphische Benutzeroberfläche 10, die in einem ersten Schritt auf graphischem Wege die Auswahl der an einer Kommunikation beteiligten Instanzen ermöglicht. Graphisches Auswählen im Zusammenhang mit der vorliegenden Erfindung bedeutet, daß ein Symbol oder ein Textvorschlag graphisch auf einer graphischen Benutzeroberfläche, beispielsweise einem PC-Bildschirm, angezeigt wird und durch schlichtes Aktivieren, d. h. beispielsweise durch Anklicken mit einer Maus, ausgewählt werden kann. Eine der Instanzen ist ein Protokolltester, auf dem das erfindungsgemäße Verfahren zur Verfügung gestellt wird, wobei der Protokolltester im vorliegenden Fall eine Komponente TC_1 emuliert. Mit den beiden Icons "Add" 12 und "Delete" 14 kann der Benutzer weitere Instanzen hinzufügen beziehungsweise aufgelistete Instanzen löschen. In einem Feld 16 ist die Zusammenstellung der Instanzen aufgelistet, während sie in einem Feld 18 als Diagramm angezeigt wird. In einem Feld 19 kann der Name der Instanz, in Feld 20 der Typ der Instanz festgelegt werden. Zwei Icons 22, 24 ermöglichen dem Benutzer, sich von einer Stufe der Definition des Kommunikationsablaufs zur nächsten zu bewegen, sowohl in Richtung auf detailliertere Spezifikation sowie auch in Richtung übergeordneter Darstellung. Eine "Cancel"-Taste 26 ermöglicht das Verlassen einer Stufe, wobei die vorgenommenen Änderungen rückgängig gemacht werden. Eine "Help"-Taste 28 bietet dem Benutzer weitere Unterstützung an.

[0017] Gemäß Figur 2, in der eine weitere Darstellung der Benutzeroberfläche 10 gezeigt ist, hat der vorliegende Kommunikationsablauf den Namen Gateway_1, siehe Feld 22. An ihm nimmt teil eine erste Instanz TC_1, gemäß Feld 24, sowie eine zweite Instanz IUT_1, gemäß Feld 26. Das emulierte Protokoll ist gemäß Feld 28a vom Typ isdnl2, wobei in Feld 28b weitere, zur Auswahl stehende Protokolle angeboten werden. In einem Feld 30 können verschiedene, zur weiteren Bearbeitung auswählbare Kommunikationsabläufe angeboten werden. Die mit Bezug auf Figur 1 beschriebenen Icons 12, 14, 22, 24, 26, 28 finden sich in vergleichbarer Form mit vergleichbarer Funktion wieder, siehe auch die noch folgenden Benutzeroberflächen, und werden daher nicht nochmals beschrieben.

[0018] Figur 3 zeigt die Benutzeroberfläche 10 von Figur 1 in einem anderen Darstellungsmodus, und zwar zur Auswahl eines SAPs, siehe Feld 32a. Im Feld 32b werden weitere SAPs zur Auswahl angeboten. Alle in Feld 32b dargestellten SAPs werden zu der gewählten Emulation isdnl2 angeboten.

[0019] Figur 4 zeigt eine weitere Darstellungsform der Benutzeroberfläche 10 von Figur 2, wobei nunmehr in einem Feld 34 aus sogenannten Message Pools ein Format für die Kommunikationsdaten (ASPs, PDUs) verwendet wird.

[0020] Figur 5 zeigt eine weitere Benutzeroberfläche 36, die in einem Feld 38 dem Benutzer diverse Informationen zur Verfügung stellt: Zum einen die von ihm ausgewählten Instanzen, dann das gemäß den Figuren 1 bis 4 vereinbarte Testszenario (Gateway_1) sowie das Datenformat (Message Pools). Im folgenden soll zunächst auf Figur 6 Bezug genommen werden: Die Benutzeroberfläche 36 weist eine Vielzahl von Icons 40 auf, die, wie von Programmen zur Textverarbeitung oder Graphikbearbeitung bekannt, beispielsweise unter Verwendung einer Maus anklickbar sind. Unter Verwendung dieser Icons läßt sich in einem Feld 42 in graphischer Weise ein Kommunikationsablauf erstellen. Figur 6 zeigt die Möglichkeit des Einbaus von Code in der Programmiersprache Forth (Draft Proposal ANSI Standard 1994) in einem Block TE_cfg 44, unter Verwendung einer Eingabemaske 46. Für die Eingabe von Code in einer anderen

Programmiersprache können weitere Eingabemasken vorgesehen werden.

[0021] Zurück zu Figur 5: Hier wird als Beispiel für einen Abschnitt eines Kommunikationsablaufs in Feld 48 dargestellt, wie zunächst alternativ die ASPs DL_ESTABLISH_CNF oder DL_ESTABLISH_IND von einer Instanz erwartet werden. Daraufhin wird ein Timer T_WaitInit der Länge 5s gestartet und das Ablaufen des Timers abgewartet.

[0022] Figur 7 zeigt, wie eine isdn-PDU "SETUP_1" als Sendenachricht in das graphisch erstellte Ablaufdiagramm eingefügt wird. In einer Eingabemaske 50 werden ASPs mit PDUs aus dem zuvor ausgewählten Message Pool angeboten. In einem optisch hervorgehobenen Feld 52 kann die ausgewählte PDU eingetragen werden. In einem Feld 54 wird weitere Information zur ausgewählten ASP beziehungsweise PDU dem Benutzer dargeboten.

[0023] Auf die soeben beschriebene Art und Weise läßt sich somit ein Kommunikationsablauf erstellen, wobei bevorzugt allen auswählbaren Parametern Beschreibungsdateien zugeordnet sind, die zum automatischen Erstellen eines zwischen den Instanzen ausführbaren Kommunikationsablaufs durch den Protokolltester automatisch miteinander verwendbar sind.

[0024] Bei der Erzeugung eines ausführbaren Codes wirken drei Komponenten zusammen: Zunächst die graphischen Benutzeroberflächen, die die gewählten Parameter, insbesondere auch die Kommunikationsabfolge in einer internen Struktur speichert. Dann ein Compiler, der die gewählten Parameter in temporäre Files übersetzt und schließlich ein Linker, der dies temporären Files liest und in die gewählte Interpreter-Scriptsprache, beispielsweise ANS Forth, umsetzt. Dabei wird der gesamte Kommunikationsablauf wie er vom Benutzer definiert wurde in ein Scriptfile geschrieben.

[0025] Im Anhang A1 ist passend zu den beschriebenen Figuren der automatisch vom Protokolltester generierte Code wiedergegeben.

Anhang A1

```
( ***** Tektronix MSC-Linker <V0.92.0> builds scenario 'isdn_user' **-*- forth
-*-** )

" $MSC$Script$" find [if] forget $MSC$Script$ [then] drop variable
$MSC$Script$
" emul"   find 0= [if] loadm emul   [then] drop
" error"  find 0= [if] loadm error  [then] drop
" mbslib" find 0= [if] loadm mbslib [then] drop
" mforth" find 0= [if] loadm mforth [then] drop
default-order
v_trace
v_screen

( >>>>>>>>>> Allocation <<<<<<<<<< )
( create instance variables and constants... )
CREATE $MSC$_InstanceVars 4 ALLOT


1 CONSTANT MSC_NUM_OF_INSTANCES
$MSC$_InstanceVars MSC_NUM_OF_INSTANCES 4 * 0 FILL
( create timer variables and constants... )
CREATE $MSC$_TimerVars 40 ALLOT
$MSC$_TimerVars 40 0 FILL
CREATE MSC_TIMER 20 ALLOT
MSC_TIMER 20 0 FILL
5 CONSTANT MSC_NUM_OF_TIMERS
( create pool variables and constants... )
CREATE $MSC$_PoolVars 4 ALLOT
1 CONSTANT MSC_NUM_OF_POOLS
$MSC$_PoolVars 4 0 FILL
( create message variables and constants... )
CREATE $MSC$_MsgVars 132 ALLOT
$MSC$_MsgVars 132 0 FILL
11 CONSTANT MSC_NUM_OF_MESSAGES
CREATE $MSC$_MsgDecodeVars 4 ALLOT
$MSC$_MsgDecodeVars 4 0 FILL
1 CONSTANT MSC_NUM_OF_MSGDECODEVARS  ( one per TM )
CREATE $MSC$_MsgFolderVars 44 ALLOT
$MSC$_MsgFolderVars 44 0 FILL
11 CONSTANT MSC_NUM_OF_FOLDERS
CREATE $MSC$_EventStructureVars MSC_NUM_OF_POOLS MSC_NUM_OF_INSTANCES * 4 *
ALLOT
$MSC$_EventStructureVars MSC_NUM_OF_POOLS MSC_NUM_OF_INSTANCES * 4 * 0 FILL
CREATE $MSC$_MsgSizeVars 4 ALLOT
$MSC$_MsgSizeVars 4 0 FILL
variable $MSC$_MsgMatched?
( create temporary variables and constants... )
variable $MSC$_TempFolderHandle
variable $MSC$_PDecoutVar
( create startstate variables... )
```

```
variable $MSC$_Req-State

( >>>>>>>>>> Constants <<<<<<<<<< )
( create mapping of gateway name to poolindex )
0 constant MSC-GW-Gateway_1 \ Mapping Gatewayname 'Gateway_1' -> Poolindex '0'

( >>>>>>>>>> Variables <<<<<<<<<< )
variable MSC-VAR-Gateway_1-SAPI
variable MSC-VAR-Gateway_1-TEI

( >>>>>>>>>> Commands <<<<<<<<<< )
include pc:boot:/share/pfe/msc_lib.4th

( constructor word ... )
: $MSC$_Constructor ( -- )
   " Cannot open pool 'pc:c:/k1297/MBS-Pools/STK-etsi93-pool1.pdc'" "
pc:c:/k1297/MBS-Pools/STK-etsi93-pool1.pdc" 0 $MSC$_OpenPool \ open pool
'pc:c:/k1297/MBS-Pools/STK-etsi93-pool1.pdc'
   " Cannot open folder 'PROT<ETSI> send to EMUL<isdnl2>'" " PROT<ETSI> send
to EMUL<isdnl2>" 0 8 $MSC$_OpenFolder \ open folder 'PROT<ETSI> send to
EMUL<isdnl2>' within pool 'pc:c:/k1297/MBS-Pools/STK-etsi93-pool1.pdc'
   " Cannot init message 'rCONN_1'" " rCONN_1" 0 8 8 $MSC$_InitMsgVar \ init
message 'rCONN_1' of pool 'pc:c:/k1297/MBS-Pools/STK-etsi93-pool1.pdc'


   " Cannot open folder  'PROT<ETSI> send to EMUL<isdnl2>'" " PROT<ETSI> send
to EMUL<isdnl2>" 0 3 $MSC$_OpenFolder \ open folder 'PROT<ETSI> send to
EMUL<isdnl2>' within pool 'pc:c:/k1297/MBS-Pools/STK-etsi93-pool1.pdc'
   " Cannot init message 'rDL_ESTABLISH_CNF_1'" " rDL_ESTABLISH_CNF_1" 0 3 3
$MSC$_InitMsgVar \ init message 'rDL_ESTABLISH_CNF_1' of pool
'pc:c:/k1297/MBS-Pools/STK-etsi93-pool1.pdc'
   " Cannot open folder 'PROT<ETSI> send to EMUL<isdnl2>'" " PROT<ETSI> send
to EMUL<isdnl2>" 0 2 $MSC$_OpenFolder \ open folder 'PROT<ETSI> send to
EMUL<isdnl2>' within pool 'pc:c:/k1297/MBS-Pools/STK-etsi93-pool1.pdc'
   " Cannot init message 'sDL_ESTABLISH_REQ_1'" " sDL_ESTABLISH_REQ_1" 0 2 2
$MSC$_InitMsgVar \ init message 'sDL_ESTABLISH_REQ_1' of pool
'pc:c:/k1297/MBS-Pools/STK-etsi93-pool1.pdc'
   " Cannot open folder 'PROT<ETSI> send to EMUL<isdnl2>'" " PROT<ETSI> send
to EMUL<isdnl2>" 0 4 $MSC$_OpenFolder \ open folder 'PROT<ETSI> send to
EMUL<isdnl2>' within pool 'pc:c:/k1297/MBS-Pools/STK-etsi93-pool1.pdc'
   " Cannot init message 'rDL_ESTABLISH_IND_1'" " rDL_ESTABLISH_IND_1" 0 4 4
$MSC$_InitMsgVar \ init message 'rDL_ESTABLISH_IND_1' of pool
'pc:c:/k1297/MBS-Pools/STK-etsi93-pool1.pdc'
   " Cannot open folder 'PROT<ETSI> send to EMUL<isdnl2>'" " PROT<ETSI> send
to EMUL<isdnl2>" 0 10 $MSC$_OpenFolder \ open folder 'PROT<ETSI> send to
EMUL<isdnl2>' within pool 'pc:c:/k1297/MBS-Pools/STK-etsi93-pool1.pdc'
   " Cannot init message 'rREL_COM_1'" " rREL_COM_1" 0 10 10 $MSC$_InitMsgVar
\ init message 'rREL_COM_1' of pool 'pc:c:/k1297/MBS-Pools/STK-etsi93-
pool1.pdc'
   " Cannot open folder 'PROT<ETSI> send to EMUL<isdnl2>'" " PROT<ETSI> send
to EMUL<isdnl2>" 0 7 $MSC$_OpenFolder \ open folder 'PROT<ETSI> send to
EMUL<isdnl2>' within pool 'pc:c:/k1297/MBS-Pools/STK-etsi93-pool1.pdc'
   " Cannot init message 'rALERT_1'" " rALERT_1" 0 7 7 $MSC$_InitMsgVar \
```

```
init message 'rALERT_1' of pool 'pc:c:/k1297/MBS-Pools/STK-etsi93-pool1.pdc'
   " Cannot open folder 'PROT<ETSI> send to EMUL<isdnl2>'" " PROT<ETSI> send
to EMUL<isdnl2>" 0 9 $MSC$_OpenFolder \ open folder 'PROT<ETSI> send to
EMUL<isdnl2>' within pool 'pc:c:/k1297/MBS-Pools/STK-etsi93-pool1.pdc'
   " Cannot init message 'sDISC_1'" " sDISC_1" 0 9 9 $MSC$_InitMsgVar \ init
message 'sDISC_1' of pool 'pc:c:/k1297/MBS-Pools/STK-etsi93-pool1.pdc'
   " Cannot open folder 'PROT<ETSI> send to EMUL<isdnl2>'" " PROT<ETSI> send
to EMUL<isdnl2>" 0 6 $MSC$_OpenFolder \ open folder 'PROT<ETSI> send to
EMUL<isdnl2>' within pool 'pc:c:/k1297/MBS-Pools/STK-etsi93-pool1.pdc'
   " Cannot init message 'rCALL_PROC_1'" " rCALL_PROC_1" 0 6 6
$MSC$_InitMsgVar \ init message 'rCALL_PROC_1' of pool 'pc:c:/k1297/MBS-
Pools/STK-etsi93-pool1.pdc'
   " Cannot open folder 'PROT<ETSI> send to EMUL<isdnl2>'" " PROT<ETSI> send
to EMUL<isdnl2>" 0 1 $MSC$_OpenFolder \ open folder 'PROT<ETSI> send to
EMUL<isdnl2>' within pool 'pc:c:/k1297/MBS-Pools/STK-etsi93-pool1.pdc'
   " Cannot init message 'rMDL_ASSIGN_CNF_1'" " rMDL_ASSIGN_CNF_1" 0 1 1
$MSC$_InitMsgVar \ init message 'rMDL_ASSIGN_CNF_1' of pool 'pc:c:/k1297/MBS-
Pools/STK-etsi93-pool1.pdc'
   " Cannot open folder 'PROT<ETSI> send to EMUL<isdnl2>'" " PROT<ETSI> send
to EMUL<isdnl2>" 0 0 $MSC$_OpenFolder \ open folder 'PROT<ETSI> send to
EMUL<isdnl2>' within pool 'pc:c:/k1297/MBS-Pools/STK-etsi93-pool1.pdc'
   " Cannot init message 'sMDL_ASSIGN_REQ_1'" " sMDL_ASSIGN_REQ_1" 0 0 0
$MSC$_InitMsgVar \ init message 'sMDL_ASSIGN_REQ_1' of pool 'pc:c:/k1297/MBS-
Pools/STK-etsi93-pool1.pdc'
   " Cannot open folder 'PROT<ETSI> send to EMUL<isdnl2>'" " PROT<ETSI> send
to EMUL<isdnl2>" 0 5 $MSC$_OpenFolder \ open folder 'PROT<ETSI> send to
EMUL<isdnl2>' within pool 'pc:c:/k1297/MBS-Pools/STK-etsi93-pool1.pdc'


   " Cannot init message 'sSETUP_1'" " sSETUP_1" 0 5 5 $MSC$_InitMsgVar \
init message 'sSETUP_1' of pool 'pc:c:/k1297/MBS-Pools/STK-etsi93-pool1.pdc'
   MSC-VAR-Gateway_1-SAPI " SAPI" " PROT<ETSI> send to EMUL<isdnl2>" 0
$MSC$_AssignMSCVar
   MSC-VAR-Gateway_1-TEI " TEI" " PROT<ETSI> send to EMUL<isdnl2>" 0
$MSC$_AssignMSCVar
   ;

( destructor word ... )
: $MSC$_Destructor ( -- )
   1 0 DO
      I $MSC$_GetPoolHandle k12mbspoolclose DROP
   LOOP
   ;

( >>>>>>>>>> Initialization <<<<<<<<<< )
5000 4 $MSC$_SetExtTimerVar \ init. timer 'T_Pause' of instance 'Phone'
45000 1 $MSC$_SetExtTimerVar \ init. timer 'T310' of instance 'Phone'
4000 0 $MSC$_SetExtTimerVar \ init. timer 'T303' of instance 'Phone'
30000 3 $MSC$_SetExtTimerVar \ init. timer 'T305' of instance 'Phone'
20000 2 $MSC$_SetExtTimerVar \ init. timer 'T_Call' of instance 'Phone'
0 0 $MSC$_InitMsg \ Create k12MBSevent structure for instance 'Phone' and
gateway 'Gateway_1'
```

```
TM0 ( >>>>>>>>>> start of instance 'Phone' <<<<<<<<<< )

( Segments of Instance 'Phone':
+--------+------------------------+------------+-------------+
|  Type  | Segment Name           | State      | Length      |
+--------+------------------------+------------+-------------+
| INIT   | - no name -            | 0000000000 | 0000000001  |
| END    | - no name -            | 0000000001 | 0000000001  |
| DOC    | START                  | 0000000002 | 0000000004  |
| DOC    | NULL                   | 0000000006 | 0000000001  |
| DOC    | CALL_INITIATED         | 0000000007 | 0000000003  |
| DOC    | CALL_PROCEEDING        | 0000000010 | 0000000003  |
| DOC    | CALL_DELIVERED_ACTIV   | 0000000013 | 0000000002  |
| DOC    | DISCONNECT_REQUEST     | 0000000015 | 0000000004  |
| CONN   | NULL                   | 0000000019 | 0000000001  |
| CONN   | CALL_INITIATED         | 0000000020 | 0000000001  |
| CONN   | CALL_PROCEEDING        | 0000000021 | 0000000001  |
| CONN   | CALL_DELIVERED_ACTIV   | 0000000022 | 0000000001  |
+--------+------------------------+------------+-------------+ )

    \ -----  init segment  -----
    0 STATE_INIT{
            128 " TM0 starts" $MSC$_TraceMsg
            0 $MSC$_ResetGotoModifierFlag \ init. instance 'Phone'
            4 $MSC$_InitTimerVar \ init. timer 'T_Pause'
            1 $MSC$_InitTimerVar \ init. timer 'T310'
            0 $MSC$_InitTimerVar \ init. timer 'T303'
            3 $MSC$_InitTimerVar \ init. timer 'T305'
            2 $MSC$_InitTimerVar \ init. timer 'T_Call'
            ( switch command for startstate... )
            $MSC$_Req-State @ CASE
              1 OF 2 NEW_STATE ENDOF


            2 OF 6 NEW_STATE ENDOF
            2 0 $MSC$_NewState ( goto START )
          ENDCASE
    }STATE_INIT

    \  ----- end segment -----
    1 STATE_INIT{
          " instance 'Phone' stops" $MSC$_PrintString
          128 " TM0 stops" $MSC$_TraceMsg
    }STATE_INIT
    1 STATE{
          ( this is the end state - loop forever )
    }STATE

    \  ----- document segment 'START' -----
    2 STATE_INIT{
          32 " Forthbox TE_cfg start " $MSC$_TraceMsg
           ( start forth box 'TE_cfg' ) " config lapd.General.Side=TE_PM"
```

```
EMU_ADMIN ( end forth box 'TE_cfg' )
          64 " Forthbox TE_cfg end " $MSC$_TraceMsg
          16 " Send message 'PROT<ETSI> send to EMUL<isdnl2>/sMDL_ASSIGN_REQ_1'
over gateway 'Gateway_1 ' " $MSC$_TraceMsg
          " Cannot send message 'sMDL_ASSIGN_REQ_1'" 0 0 0 $MSC$_SendPrimitive
   }STATE_INIT
   2 STATE{
       " Error while matching primitive 'rMDL_ASSIGN_CNF_1'" 1 0 0
$MSC$_RecvPrimitive
       ACTION{
          8 " Received message 'PROT<ETSI> send to
EMUL<isdnl2>/rMDL_ASSIGN_CNF_1' from gateway 'Gateway_1 ' " $MSC$_TraceMsg
          0 0 1 $MSC$_FreeEventStructure \ free event structure of message
'PROT<ETSI> send to EMUL<isdnl2>/rMDL_ASSIGN_CNF_1' and gateway 'Gateway_1'
          1 $MSC$_ResetMsgFlag \ message 'PROT<ETSI> send to
EMUL<isdnl2>/rMDL_ASSIGN_CNF_1' from gateway 'Gateway_1'
          0 $MSC$_ResetGotoModifierFlag
          16 " Send message 'PROT<ETSI> send to
EMUL<isdnl2>/sDL_ESTABLISH_REQ_1' over gateway 'Gateway_1 ' " $MSC$_TraceMsg
          " Cannot send message 'sDL_ESTABLISH_REQ_1'" 2 0 0
$MSC$_SendPrimitive
          3 0 $MSC$_NewState
       }ACTION
       ?TM_TIMEOUT
       ACTION{
          1 " Unexpected timer event " $MSC$_TraceMsg
       }ACTION
       FALSE E-SAP @ 0 = OR
       ACTION{
          8 " Unexpected message event " $MSC$_TraceMsg
          0 0 1 $MSC$_FreeEventStructure \ free event structure of message
'PROT<ETSI> send to EMUL<isdnl2>/rMDL_ASSIGN_CNF_1' and gateway 'Gateway_1'
       }ACTION
   }STATE
   3 STATE_INIT{
          3 $MSC$_ResetMsgFlag \ message 'PROT<ETSI> send to
EMUL<isdnl2>/rDL_ESTABLISH_CNF_1' from gateway 'Gateway_1'


   4 $MSC$_ResetMsgFlag \ message 'PROT<ETSI> send to
 EMUL<isdnl2>/rDL_ESTABLISH_IND_1' from gateway 'Gateway_1'
    }STATE_INIT
   3 STATE{
       " Error while matching primitive 'rDL_ESTABLISH_CNF_1'" 3 0 0
 $MSC$_RecvPrimitive
       ACTION{
          0 0 3 $MSC$_FreeEventStructure \ free event structure of message
 'PROT<ETSI> send to EMUL<isdnl2>/rDL_ESTABLISH_CNF_1' and gateway 'Gateway_1'
          0 0 4 $MSC$_FreeEventStructure \ free event structure of message
 'PROT<ETSI> send to EMUL<isdnl2>/rDL_ESTABLISH_IND_1' and gateway 'Gateway_1'
          0 $MSC$_SetGotoModifierFlag
          4 0 $MSC$_NewState
```

```
        }ACTION
        " Error while matching primitive 'rDL_ESTABLISH_IND_1'" 4 0 0
    $MSC$_RecvPrimitive
        ACTION{
          0 0 3 $MSC$_FreeEventStructure \ free event structure of message
    'PROT<ETSI> send to EMUL<isdnl2>/rDL_ESTABLISH_CNF_1' and gateway 'Gateway_1'
          0 0 4 $MSC$_FreeEventStructure \ free event structure of message
    'PROT<ETSI> send to EMUL<isdnl2>/rDL_ESTABLISH_IND_1' and gateway 'Gateway_1'
          0 $MSC$_SetGotoModifierFlag
          5 0 $MSC$_NewState
        }ACTION
        ?TM_TIMEOUT
        ACTION{
          1 " Unexpected timer event " $MSC$_TraceMsg
          0 0 3 $MSC$_FreeEventStructure \ free event structure of message
    'PROT<ETSI> send to EMUL<isdnl2>/rDL_ESTABLISH_CNF_1' and gateway 'Gateway_1'
          0 0 4 $MSC$_FreeEventStructure \ free event structure of message
    'PROT<ETSI> send to EMUL<isdnl2>/rDL_ESTABLISH_IND_1' and gateway 'Gateway_1'
        }ACTION
        FALSE E-SAP @ 0 = OR
        ACTION{
          8 " Unexpected message event " $MSC$_TraceMsg
          0 0 3 $MSC$_FreeEventStructure \ free event structure of dessage
    'PROT<ETSI> send to EMUL<isdnl2>/rDL_ESTABLISH_CNF_1' and gateway 'Gateway_1'
          0 0 4 $MSC$_FreeEventStructure \ free event structure of message
    'PROT<ETSI> send to EMUL<isdnl2>/rDL_ESTABLISH_IND_1' and gateway 'Gateway_1'
        }ACTION
      }STATE
      4 STATE{
        " Error while matching primitive 'rDL_ESTABLISH_CNF_1'" 3 0 0
    $MSC$_RecvPrimitive
        ACTION{
          8 " Received message 'PROT<ETSI> send to
    EMUL<isdnl2>/rDL_ESTABLISH_CNF_1' from gateway 'Gateway_1 ' " $MSC$_TraceMsg
          0 0 3 $MSC$_FreeEventStructure \ free event structure of message
    'PROT<ETSI> send to EMUL<isdnl2>/rDL_ESTABLISH_CNF_1' and gateway 'Gateway_1'
          3 $MSC$_ResetMsgFlag \ message 'PROT<ETSI> send to
    EMUL<isdnl2>/rDL_ESTABLISH_CNF_1' from gateway 'Gateway_1'
          0 $MSC$_ResetGotoModifierFlag
          6 0 $MSC$_NewState
        }ACTION
        ?TM_TIMEOUT
        ACTION{
          1 " Unexpected timer event " $MSC$_TraceMsg
        }ACTION
        FALSE E-SAP @ 0 = OR
        ACTION{
          8 " unexpected message event " $MSC$_TraceMsg
          0 0 3 $MSC$_FreeEventStructure \ free event structure of message
    'PROT<ETSI> send to EMUL<isdnl2>/rDL_ESTABLISH_CNF_1' and gateway 'Gateway_1'
        }ACTION
      }STATE
      5 STATE{
```

```
      " Error while matching primitive 'rDL_ESTABLISH_IND_1'" 4 0 0
$MSC$_RecvPrimitive
       ACTION{
           8 " Received message 'PROT<ETSI> send to
EMUL<isdnl2>/rDL_ESTABLISH_IND_1' from gateway 'Gateway_1 ' " $MSC$_TraceMsg
           0 0 4 $MSC$_FreeEventStructure \ free event structure of message
'PROT<ETSI> send to EMUL<isdnl2>/rDL_ESTABLISH_IND_1' and gateway 'Gateway_1'
           4 $MSC$_ResetMsgFlag \ message 'PROT<ETSI> send to
EMUL<isdnl2>/rDL_ESTABLISH_IND_1' from gateway 'Gateway_1'
           0 $MSC$_ResetGotoModifierFlag
           6 0 $MSC$_NewState
       }ACTION
       ?TM_TIMEOUT
       ACTION{
           1 " Unexpected timer event " $MSC$_TraceMsg
       }ACTION
       FALSE E-SAP @ 0 = OR
       ACTION{
           8 " Unexpected message event " $MSC$_TraceMsg
           0 0 4 $MSC$_FreeEventStructure \ free event structure of message
'PROT<ETSI> send to EMUL<isdnl2>/rDL_ESTABLISH_IND_1' and gateway 'Gateway_1'
       }ACTION
    }STATE

       \  ----- document segment 'NULL' -----
   6 STATE_INIT{
           16 " Send message 'PROT<ETSI> send to EMUL<isdnl2>/sSETUP_1' over
gateway 'Gateway_1 ' " $MSC$_TraceMsg
           " Cannot send message 'sSETUP_1'" 5 0 0 $MSC$_SendPrimitive
           2 " Timer 'T303' set with value '4000'" $MSC$_TraceMsg
           4000 0 $MSC$_SetTimer \ timer 'T303'
           19 0 $MSC$_NewState
   }STATE_INIT

       \  ----- document segment 'CALL_INITIATED' -----
   7 STATE{
       " Error while matching primitive 'rCALL_PROC_1'" 6 0 0
$MSC$_RecvPrimitive
       ACTION{
           0 0 6 $MSC$_FreeEventStructure \ free event structure of message
'PROT<ETSI> send to EMUL<isdnl2>/rCALL_PROC_1' and gateway 'Gateway_1'
           0 $MSC$_SetGotoModifierFlag
           8 0 $MSC$_NewState
       }ACTION
       0 $MSC$_Timeout \ timer 'T303'
       ACTION{

           0 0 6 $MSC$_FreeEventStructure \ free event structure of message
'PROT<ETSI> send to EMUL<isdnl2>/rCALL_PROC_1' and gateway 'Gateway_1'
           0 $MSC$_SetGotoModifierFlag
           9 0 $MSC$_NewState
```

```
        }ACTION
        ?TM_TIMEOUT
        ACTION{
           1 " Unexpected timer event " $MSC$_TraceMsg
           0 0 6 $MSC$_FreeEventStructure \ free event structure of message
'PROT<ETSI> send to EMUL<isdnl2>/rCALL_PROC_1' and gateway 'Gateway_1'
        }ACTION
        FALSE E-SAP @ 0 = OR
        ACTION{
           8 " Unexpected message event " $MSC$_TraceMsg
           0 0 6 $MSC$_FreeEventStructure \ free event structure of message
'PROT<ETSI> send to EMUL<isdnl2>/rCALL_PROC_1' and gateway 'Gateway_1'
        }ACTION
     }STATE
    8 STATE{
        " Error while matching primitive 'rCALL_PROC_1'" 6 0 0
$MSC$_RecvPrimitive
        ACTION{
           8 " Received message 'PROT<ETSI> send to EMUL<isdnl2>/rCALL_PROC_1'
from gateway 'Gateway_1 ' " $MSC$_TraceMsg
           0 0 6 $MSC$_FreeEventStructure \ free event structure of message
'PROT<ETSI> send to EMUL<isdnl2>/rCALL_PROC_1' and gateway 'Gateway_1'
           6 $MSC$_ResetMsgFlag \ message 'PROT<ETSI> send tc
EMUL<isdnl2>/rCALL_PROC_1' from gateway 'Gateway_1'
           0 $MSC$_ResetGotoModifierFlag
           4 " Timer 'T303' reset" $MSC$_TraceMsg
           0 $MSC$_ResetTimer \ timer 'T303'
           2 " Timer 'T310' set with value '45000'" $MSC$_TraceMsg
           45000 1 $MSC$_SetTimer \ timer 'T310'
           20 0 $MSC$_NewState
        }ACTION
        ?TM_TIMEOUT
        ACTION{
           1 " Unexpected timer event " $MSC$_TraceMsg
        }ACTION
        FALSE E-SAP @ 0 = OR
        ACTION{
           8 " Unexpected message event " $MSC$_TraceMsg
           0 0 6 $MSC$_FreeEventStructure \ free event structure of message
 'PROT<ETSI> send to EMUL<isdnl2>/rCALL_PROC_1' and gateway 'Gateway_1'
        }ACTION
     }STATE
    9 STATE{
        0 $MSC$_Timeout \ timer 'T303'
        ACTION{
           1 " Received timeout 'T303' " $MSC$_TraceMsg
           0 $MSC$_ResetTimerFlag \ timer 'T303'
           0 $MSC$_ResetGotoModifierFlag
           20 0 $MSC$_NewState
        }ACTION
        ?TM_TIMEOUT
        ACTION{
```

```
            1 " Unexpected timer event " $MSC$_TraceMsg
         }ACTION
         FALSE E-SAP @ 0 - OR
         ACTION{
            8 " Unexpected message event " $MSC$_TraceMsg
         }ACTION
      }STATE

      \  ----- document segment 'CALL_PROCEEDING'  -----
      10 STATE{
         " Error while matching primitive 'rALERT_1'" 7 0 0 $MSC$_RecvPrimitive
         ACTION{
            0 0 7 $MSC$_FreeEventStructure \ free event structure of message
'PROT<ETSI> send to EMUL<isdnl2>/rALERT_1' and gateway 'Gateway_1'
            0 $MSC$_SetGotoModifierFlag
            11 0 $MSC$_NewState
         }ACTION
         1 $MSC$_Timeout \ timer 'T310'
         ACTION{
            0 0 7 $MSC$_FreeEventStructure \ free event structure of message
'PROT<ETSI> send to EMUL<isdnl2>/rALERT_1' and gateway 'Gateway_1'
            0 $MSC$_SetGotoModifierFlag
            12 0 $MSC$_NewState
         }ACTION
         ?TM_TIMEOUT
         ACTION{
            1 " Unexpected timer event " $MSC$_TraceMsg
            0 0 7 $MSC$_FreeEventStructure \ free event structure of message
'PROT<ETSI> send to EMUL<isdnl2>/rALERT_1' and gateway 'Gateway_1'
         }ACTION
         FALSE E-SAP @ 0 - OR
         ACTION{
            8 " Unexpected message event " $MSC$_TraceMsg
            0 0 7 $MSC$_FreeEventStructure \ free event structure of message
'PROT<ETSI> send to EMUL<isdnl2>/rALERT_1' and gateway 'Gateway_1'
         }ACTION
      }STATE
      11 STATE{
         " Error while matching primitive 'rALERT_1' " 7 0 0 $MSC$_RecvPrimitive
         ACTION{
            8 " Received message 'PROT<ETSI> send to EMUL<isdnl2>/rALERT_1' from
gateway 'Gateway_1 ' " $MSC$_TraceMsg
            0 0 7 $MSC$_FreeEventStructure \ free event structure of message
'PROT<ETSI> send to EMUL<isdnl2>/rALERT_1' and gateway 'Gateway_1'
            7 $MSC$_ResetMsgFlag \ message 'PROT<ETSI> send to
EMUL<isdnl2>/rALERT_1' from gateway 'Gateway_1'
            0 $MSC$_ResetGotoModifierFlag
            4 " Timer 'T310' reset" $MSC$_TraceMsg
            1 $MSC$_ResetTimer \ timer 'T310'
            21 0 $MSC$_NewState
         }ACTION
         ?TM_TIMEOUT
```

```
        ACTION{
           1 " Unexpected timer event " $MSC$_TraceMsg
        }ACTION
        FALSE E-SAP @ 0 = OR



         ACTION{
           8 " Unexpected message event " $MSC$_TraceMsg
           0 0 7 $MSC$_FreeEventStructure \ free event structure of message
'PROT<ETSI> send to EMUL<isdnl2>/rALERT_1' and gateway 'Gateway_1'
        }ACTION
     }STATE
     12 STATE{
        1 $MSC$_Timeout \ timer 'T310'
        ACTION{
           1 " Received timeout 'T310' " $MSC$_TraceMsg
           1 $MSC$_ResetTimerFlag \ timer 'T310'
           0 $MSC$_ResetGotoModifierFlag
           21 0 $MSC$_NewState
        }ACTION
        ?TM_TIMEOUT
        ACTION{
           1 " Unexpected timer event " $MSC$_TraceMsg
        }ACTION
        FALSE E-SAP @ 0 = OR
        ACTION{
           8 " Unexpected message event " $MSC$_TraceMsg
        }ACTION
     }STATE

     \ ----- document segment 'CALL_DELIVERED_ACTIVE'   -----
     13 STATE{
        " Error while matching primitive 'rCONN_1'" 8 0 0 $MSC$_RecvPrimitive
        ACTION{
           8 " Received message 'PROT<ETSI> send to EMUL<isdnl2>/rCONN_1' from
gateway 'Gateway_1 ' " $MSC$_TraceMsg
           0 0 8 $MSC$_FreeEventStructure \ free event structure of message
'PROT<ETSI> send to EMUL<isdnl2>/rCONN_1' and gateway 'Gateway_1'
           8 $MSC$_ResetMsgFlag \ message 'PROT<ETSI> send to
EMUL<isdnl2>/rCONN_1' from gateway 'Gateway_1'
           0 $MSC$_ResetGotoModifierFlag
           2 " Timer 'T_Call' set with value '20000'" $MSC$_TraceMsg
           20000 2 $MSC$_SetTimer \ timer 'T_Call'
           14 0 $MSC$_NewState
        }ACTION
        ?TM_TIMEOUT
        ACTION{
           1 " Unexpected timer event " $MSC$_TraceMsg
        }ACTION
        FALSE E-SAP @ 0 = OR
        ACTION{
           8 " Unexpected message event " $MSC$_TraceMsg
```

```
            0 0 8 $MSC$_FreeEventStructure \ free event structure of message
'PROT<ETSI> send to EMUL<isdnl2>/rCONN_1' and gateway 'Gateway_1'
         }ACTION
      }STATE
      14 STATE{
         2 $MSC$_Timeout \ timer 'T_Call'
         ACTION{
            1 " Received timeout 'T_Call' " $MSC$_TraceMsg
            2 $MSC$_ResetTimerFlag \ timer 'T_Call'
            0 $MSC$_ResetGotoModifierFlag


            16 " Send message 'PROT<ETSI> send to EMUL<isdnl2>/sDISC_1' over
gateway 'Gateway_1 ' " $MSC$_TraceMsg
            " Cannot send message 'sDISC_1'" 9 0 0 $MSC$_SendPrimitive
            2 " Timer 'T305' set with value '30000'" $MSC$_TraceMsg
            30000 3 $MSC$_SetTimer \ timer 'T305'
            22 0 $MSC$_NewState
         }ACTION
         ?TM_TIMEOUT
         ACTION{
            1 " Unexpected timer event " $MSC$_TraceMsg
         }ACTION
         FALSE E-SAP @ 0 = OR
         ACTION{
            8 " Unexpected message event " $MSC$_TraceMsg
         }ACTION
      }STATE

      \  ----- document segment 'DISCONNECT_REQUEST' -----
      15 STATE{
         " Error while matching primitive 'rREL_COM_1'" 10 0 0
$MSC$_RecvPrimitive
         ACTION{
            0 0 10 $MSC$_FreeEventStructure \ free event structure of message
'PROT<ETSI> send to EMUL<isdnl2>/rREL_COM_1' and gateway 'Gateway_1'
            0 $MSC$_SetGotoModifierFlag
            16 0 $MSC$_NewState
         }ACTION
         3 $MSC$_Timeout \ timer 'T305'
         ACTION{
            0 0 10 $MSC$_FreeEventStructure \ free event structure of message
'PROT<ETSI> send to EMUL<isdnl2>/rREL_COM_1' and gateway 'Gateway_1'
            0 $MSC$_SetGotoModifierFlag
            17 0 $MSC$_NewState
         }ACTION
         ?TM_TIMEOUT
         ACTION{
            1 " Unexpected timer event " $MSC$_TraceMsg
            0 0 10 $MSC$_FreeEventStructure \ free event structure of message
'PROT<ETSI> send to EMUL<isdnl2>/rREL_COM_1' and gateway 'Gateway_1'
         }ACTION
         FALSE E-SAP @ 0 = OR
```

```
        ACTION{
            8 " Unexpected message event " $MSC$_TraceMsg
            0 0 10 $MSC$_FreeEventStructure \ free event structure of message
'PROT<ETSI> send to EMUL<isdnl2>/rREL_COM_1' and gateway 'Gateway_1'
        }ACTION
      }STATE
    16 STATE{
        " Error while matching primitive 'rREL_COM_1'" 10 0 0
$MSC$_RecvPrimitive
        ACTION{
            8 " Received message 'PROT<ETSI> send to EMUL<isdnl2>/rREL_COM_1'
from gateway 'Gateway_1 ' " $MSC$_TraceMsg
            0 0 10 $MSC$_FreeEventStructure \ free event structure of message
'PROT<ETSI> send to EMUL<isdnl2>/rREL_COM_1' and gateway 'Gateway_1'



            10 $MSC$_ResetMsgFlag \ message 'PROT<ETSI> send to
EMUL<isdnl2>/rREL_COM_1' from gateway 'Gateway_1'
            0 $MSC$_ResetGotoModifierFlag
            18 0 $MSC$_NewState
        }ACTION
        ?TM_TIMEOUT
        ACTION{
            1 " Unexpected timer event " $MSC$_TraceMsg
        }ACTION
        FALSE E-SAP @ 0 = OR
        ACTION{
            8 " Unexpected message event " $MSC$_TraceMsg
            0 0 10 $MSC$_FreeEventStructure \ free event structure of message
'PROT<ETSI> send to EMUL<isdnl2>/rREL_COM_1' and gateway 'Gateway_1'
        }ACTION
    }STATE
    17 STATE{
        3 $MSC$_Timeout \ timer 'T305'
        ACTION{
            1 " Received timeout 'T305' " $MSC$_TraceMsg
            3 $MSC$_ResetTimerFlag \ timer 'T305'
            0 $MSC$_ResetGotoModifierFlag
            18 0 $MSC$_NewState
        }ACTION
        ?TM_TIMEOUT
        ACTION{
            1 " Unexpected timer event " $MSC$_TraceMsg
        }ACTION
        FALSE E-SAP @ 0 = OR
        ACTION{
            8 " Unexpected message event " $MSC$_TraceMsg
        }ACTION
    }STATE
    18 STATE_INIT{
            2 " Timer 'T_Pause' set with value '5000'" $MSC$_TraceMsg
            5000 4 $MSC$_SetTimer \ timer 'T_Pause'
```

```
}STATE_INIT
18 STATE{
    4 $MSC$_Timeout \ timer 'T_Pause'
    ACTION{
       1 " Received timeout 'T_Pause' " $MSC$_TraceMsg
       4 $MSC$_ResetTimerFlag \ timer 'T_Pause'
       0 $MSC$_ResetGotoModifierFlag
       1 0 $MSC$_NewState
    }ACTION
    ?TM_TIMEOUT
    ACTION{
       1 " Unexpected timer event " $MSC$_TraceMsg
    }ACTION
    FALSE E-SAP @ 0 = OR
    ACTION{
       8 " Unexpected message event " $MSC$_TraceMsg
    }ACTION
  }STATE

   \  ----- connector segment 'NULL' -----


19 STATE_INIT{
       6 $MSC$_ResetMsgFlag \ message 'PROT<ETSI> send to
EMUL<isdnl2>/rCALL_PROC_1'
       0 $MSC$_ResetTimerFlag \ timer 'T303'
}STATE_INIT
19 STATE{
    " Error while matching primitive 'rCALL_PROC_1'" 6 0 0
 $MSC$_RecvPrimitive
    ACTION{
       0 0 6 $MSC$_FreeEventStructure \ free event structure of message
 'PROT<ETSI> send to EMUL<isdnl2>/rCALL_PROC_1' and gateway 'Gateway_1'
       0 $MSC$_SetGotoModifierFlag
       6 $MSC$_SetMsgFlag \ message 'PROT<ETSI> send to
 EMUL<isdnl2>/rCALL_PROC_1'
       7 0 $MSC$_NewState
    }ACTION
    0 $MSC$_Timeout \ timer 'T303'
    ACTION{
       0 0 6 $MSC$_FreeEventStructure \ free event structure of message
 'PROT<ETSI> send to EMUL<isdnl2>/rCALL_PROC_1' and gateway 'Gateway_1'
       0 $MSC$_SetGotoModifierFlag
       0 $MSC$_SetTimerFlag \ timer 'T303'
       7 0 $MSC$_NewState
    }ACTION
    ?TM_TIMEOUT
    ACTION{
        " Unexpected timer event" $MSC$_TraceMsg
    }ACTION
    FALSE E-SAP @ 0 = OR
    ACTION{
       " Unexpected message event" $MSC$_TraceMsg
```

```
            0 0 6 $MSC$_FreeEventStructure \ free event structure of message
'PROT<ETSI> send to EMUL<isdnl2>/rCALL_PROC_1' and gateway 'Gateway_1'
        }ACTION
    }STATE

    \  ----- connector segment 'CALL_INITIATED' -----
    20 STATE_INIT{
            7 $MSC$_ResetMsgFlag \ message 'PROT<ETSI> send to
EMUL<isdnl2>/rALERT_1'
            1 $MSC$_ResetTimerFlag \ timer 'T310'
    }STATE_INIT
    20 STATE{
        " Error while matching primitive 'rALERT_1'" 7 0 0 $MSC$_RecvPrimitive
        ACTION{
            0 0 7 $MSC$_FreeEventStructure \ free event structure of message
'PROT<ETSI> send to EMUL<isdnl2>/rALERT_1' and gateway 'Gateway_1'
            0 $MSC$_SetGotoModifierFlag
            7 $MSC$_SetMsgFlag \ message 'PROT<ETSI> send to
EMUL<isdnl2>/rALERT_1'
            10 0 $MSC$_NewState
        }ACTION
        1 $MSC$_Timeout \ timer 'T310'
        ACTION{
            0 0 7 $MSC$_FreeEventStructure \ free event structure of message
'PROT<ETSI> send to EMUL<isdnl2>/rALERT_1' and gateway 'Gateway_1'


            0 $MSC$_SetGotoModifierFlag
            1 $MSC$_SetTimerFlag \ timer 'T310'
            10 0 $MSC$_NewState
        }ACTION
        ?TM_TIMEOUT
        ACTION{
            " Unexpected timer event" $MSC$_TraceMsg
        }ACTION
        FALSE E-SAP @ 0 = OR
        ACTION{
            " Unexpected message event" $MSC$_TraceMsg
            0 0 7 $MSC$_FreeEventStructure \ free event structure of message
'PROT<ETSI> send to EMUL<isdnl2>/rALERT_1' and gateway 'Gateway_1'
        }ACTION
    }STATE

    \  ----- connector segment 'CALL_PROCEEDING' -----
    21 STATE_INIT{
            8 $MSC$_ResetMsgFlag \ message 'PROT<ETSI> send to
EMUL<isdnl2>/rCONN_1'
    }STATE_INIT
    21 STATE{
        " Error while matching primitive 'rCONN_1'" 8 0 0 $MSC$_RecvPrimitive
        ACTION{
            0 0 8 $MSC$_FreeEventStructure \ free event structure of message
'PROT<ETSI> send to EMUL<isdnl2>/rCONN_1' and gateway 'Gateway_1'
```

```
            0 $MSC$_SetGotoModifierFlag
            8 $MSC$_SetMsgFlag \ message 'PROT<ETSI> send to
EMUL<isdnl2>/rCONN_1'
            13 0 $MSC$_NewState
         }ACTION
         ?TM_TIMEOUT
         ACTION{
            " Unexpected timer event" $MSC$_TraceMsg
         }ACTION
         FALSE E-SAP @ 0 = OR
         ACTION{
            " Unexpected message event" $MSC$_TraceMsg
            0 0 8 $MSC$_FreeEventStructure \ free event structure of message
'PROT<ETSI> send to EMUL<isdnl2>/rCONN_1' and gateway 'Gateway_1'
         }ACTION
      }STATE

       \  ----- connector segment 'CALL_DELIVERED_ACTIVE' -----
      22 STATE_INIT{
            10 $MSC$_ResetMsgFlag \ message 'PROT<ETSI> send to
EMUL<isdnl2>/rREL_COM_1'
            3 $MSC$_ResetTimerFlag \ timer 'T305'
      }STATE_INIT
      22 STATE{
         " Error while matching primitive 'rREL_COM_1'" 10 0 0
$MSC$_ RecvPrimitive
         ACTION{
            0 0 10 $MSC$_FreeEventStructure \ free event structure of message
'PROT<ETSI> send to EMUL<isdnl2>/rREL_COM_1' and gateway 'Gateway_1'
            0 $MSC$_SetGotoModifierFlag


10 $MSC$_SetMsgFlag \ message 'PROT<ETSI> send to
EMUL<isdnl2>/rREL_COM_1'
            15 0 $MSC$_NewState
         }ACTION
         3 $MSC$_Timeout \ timer 'T305'
         ACTION{
            0 0 10 $MSC$_FreeEventStructure \ free event structure of message
'PROT<ETSI> send to EMUL<isdnl2>/rREL_COM_1' and gateway 'Gateway_1'
            0 $MSC$_SetGotoModifierFlag
            3 $MSC$_SetTimerFlag \ timer 'T305'
            15 0 $MSC$_NewState
         }ACTION
         ?TM_TIMEOUT
         ACTION{
            " Unexpected timer event" $MSC$_TraceMsg
         }ACTION
         FALSE E-SAP @ 0 = OR
         ACTION{
            " Unexpected message event" $MSC$_TraceMsg
            0 0 10 $MSC$_FreeEventStructure \ free event structure of message
'PROT<ETSI> send to EMUL<isdnl2>/rREL_COM_1' and gateway 'Gateway_1'
```

```
        }ACTION
      }STATE
( >>>>>>>>>> end of instance 'Phone' <<<<<<<<<< )

$MSC$_Constructor
MSC_MENU_CTRL_FCT ( calls the menu control function )
" MSC scenario 'isdn_user' loaded" $MSC$_PrintString
```

Patentansprüche

1. Verfahren zum Erstellen eines Ablaufs einer zwischen mindestens zwei Instanzen ablaufenden Kommunikation, wobei eine Instanz ein Protokolltester ist, gekennzeichnet durch folgende, am Protokolltester ausführbare Schritte:

   a) Auswählen der an der Kommunikation beteiligten Instanzen;
   b) Auswählen einer Protokollschicht, auf deren Grundlage die Kommunikation zwischen den ausgewählten Instanzen ablaufen soll;
   c) Auswählen derjenigen abstrakten Kommunikationsschnittstellen der Protokollschicht, die an der Kommunikation beteiligt sind;
   d) Auswählen der Kommunikationsdaten;
   e) automatisches Erstellen eines zwischen den mindestens zwei Instanzen ausführbaren Kommunikationsablaufs durch den Protokolltester, auf der Grundlage der Auswahlen in den Schritten a) bis d),

   wobei die Auswahl von Schritt c) und/oder Schritt d) graphisch erfolgt und den dabei auswählbaren Parametern Beschreibungsdateien zugeordnet sind, die in Schritt e) zur Erstellung eines zwischen den Instanzen ausführbaren Kommunikationsablaufs verwendet werden.

2. Verfahren nach Anspruch 1,
   dadurch gekennzeichnet,
   daß weiterhin in Schritt a) die an der Kommunikation beteiligten Instanzen graphisch ausgewählt werden und/oder in Schritt b) die Protokollschicht graphisch ausgewählt wird und den dabei auswählbaren Parametern Beschreibungsdateien zugeordnet sind, die in Schritt e) zur Erstellung eines zwischen den Instanzen ausführbaren Kommunikationsablaufs verwendet werden.

3. Verfahren nach Anspruch 1 oder 2,
   dadurch gekennzeichnet,
   daß die abstrakten Kommunikationsschnittstellen SAPs (Service Access Points) umfassen.

4. Verfahren nach einem der vorhergehenden Ansprüche,
   dadurch gekennzeichnet,
   daß die Kommunikationsdaten PDUs (Protocol Data Units) und/oder ASPs (Abstract Service Primitives) umfassen.

5. Verfahren nach einem der vorhergehenden Ansprüche,
   dadurch gekennzeichnet,
   daß Schritt d) folgende Teilschritte umfaßt:

   d1) graphisches Auswählen eines Datenformats;
   d2) graphischer Aufbau einer Kommunikationsabfolge zwischen den beteiligten Instanzen.

6. Verfahren nach Anspruch 5,
   dadurch gekennzeichnet,
   daß in Schritt d2) Source-Code eingebbar ist.

7. Verfahren nach einem der vorhergehenden Ansprüche,

dadurch g k nnz ichnet,
daß allen auswählbaren Parametern Beschreibungsdateien zugeordnet sind, die in Schritt e) zur Erstellung eines zwischen den Instanzen ausführbaren Kommunikationsablaufs verwendet werden.

8. Protokolltester mit

a) Mitteln zum Auswählen der an einer Kommunikation beteiligten Instanzen (19, 20, 24, 26), wobei eine der Instanzen der Protokolltester ist;
b) Mitteln zum Auswählen einer Protokollschicht (20, 28a), auf deren Grundlage die Kommunikation zwischen den ausgewählten Instanzen ablaufen soll;
c) Mitteln zum Auswählen derjenigen abstrakten Kommunikationsschnittstellen (20, 32a) der Protokollschicht, die an der Kommunikation beteiligt sind;
d) Mitteln zum Auswählen der Kommunikationsdaten (20, 34);
e) Mitteln zum automatischen Erstellen eines zwischen den Instanzen ausführbaren Kommunikationsablaufs durch den Protokolltester, auf der Grundlage der Auswahlen gemäß a) bis d),

wobei die Auswahlmittel gemäß c) und/oder gemäß d) graphische Auswahlmittel sind und den durch sie auswählbaren Parametern Beschreibungsdateien zugeordnet sind, die gemäß e) von den Erstellungsmitteln zur Erstellung eines zwischen den Instanzen ausführbaren Kommunikationsablaufs verwendbar sind.

9. Protokolltester nach Anspruch 8,
dadurch gekennzeichnet,
daß die Mittel zum Auswählen der an der Kommunikation beteiligten Instanzen (19, 20, 24, 26) und/oder die Mittel zum Auswählen der Protokollschicht (20, 28a) graphische Auswahlmittel sind und den durch sie auswählbaren Parametern Beschreibungsdateien zugeordnet sind, die gemäß e) von den Erstellungsmitteln zur Erstellung eines zwischen den Instanzen ausführbaren Kommunikationsablaufs verwendbar sind.

10. Protokolltester nach einem der Ansprüche 8 oder 9,
dadurch gekennzeichnet,
daß die abstrakten Kommunikationsschnittstellen SAPs (Service Access Points) umfassen.

11. Protokolltester nach einem der Ansprüche 8 bis 10,
dadurch gekennzeichnet,
daß die Kommunikationsdaten PDUs (Protocol Data Units) und/oder ASPs (Abstract Service Primitives) umfassen.

12. Protokolltester nach einem der Ansprüche 8 bis 11,
dadurch gekennzeichnet,
daß er Mittel zum Eingeben von Source-Code (44, 46) umfaßt.

13. Protokolltester nach einem der Ansprüche 8 bis 12,
dadurch gekennzeichnet,
daß allen mit den Auswahlmitteln auswählbaren Parametern Beschreibungsdateien zugeordnet sind, die gemäß e) von den Erstellungsmitteln zur Erstellung eines zwischen den Instanzen ausführbaren Kommunikationsablaufs verwendbar sind.

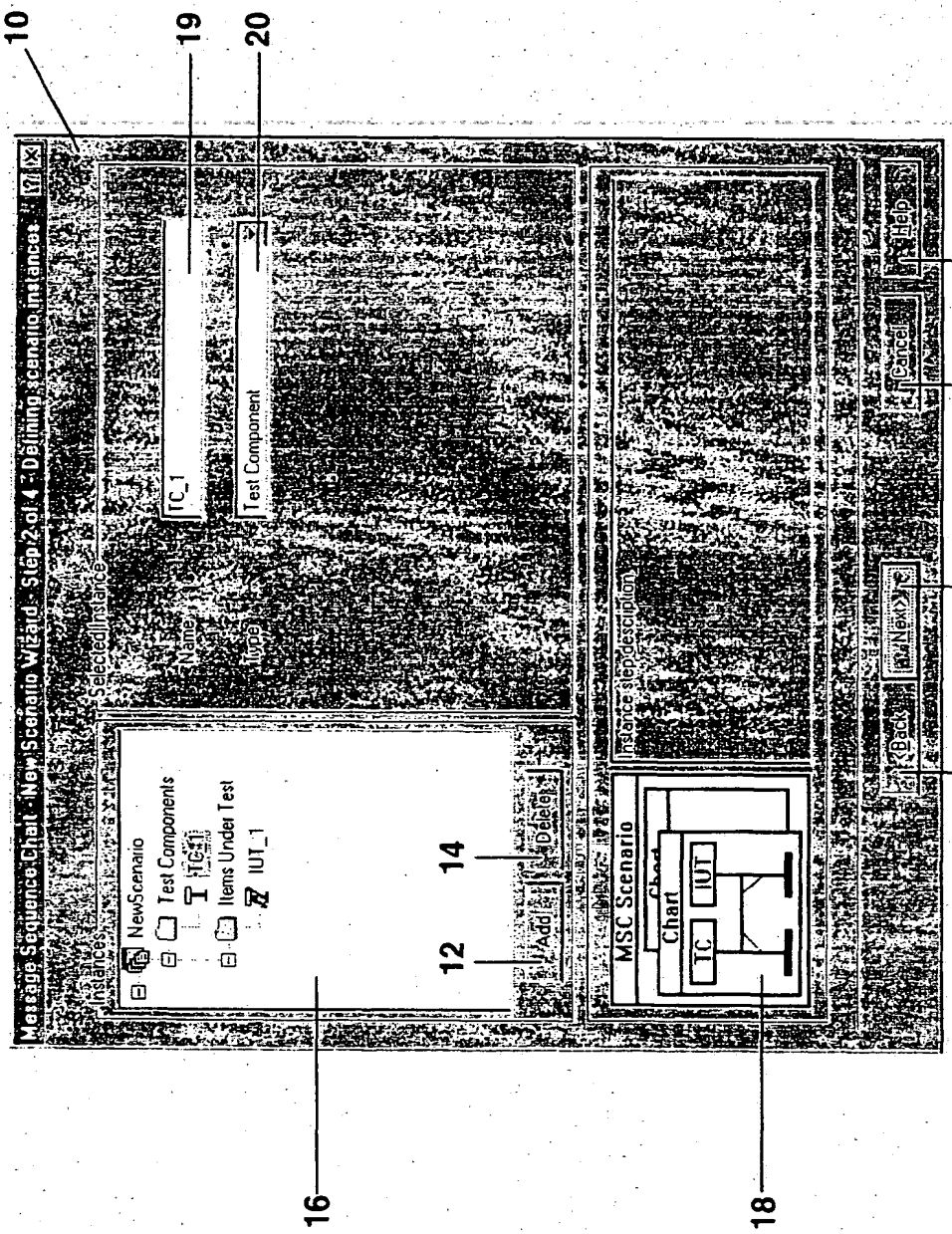


Fig. 1

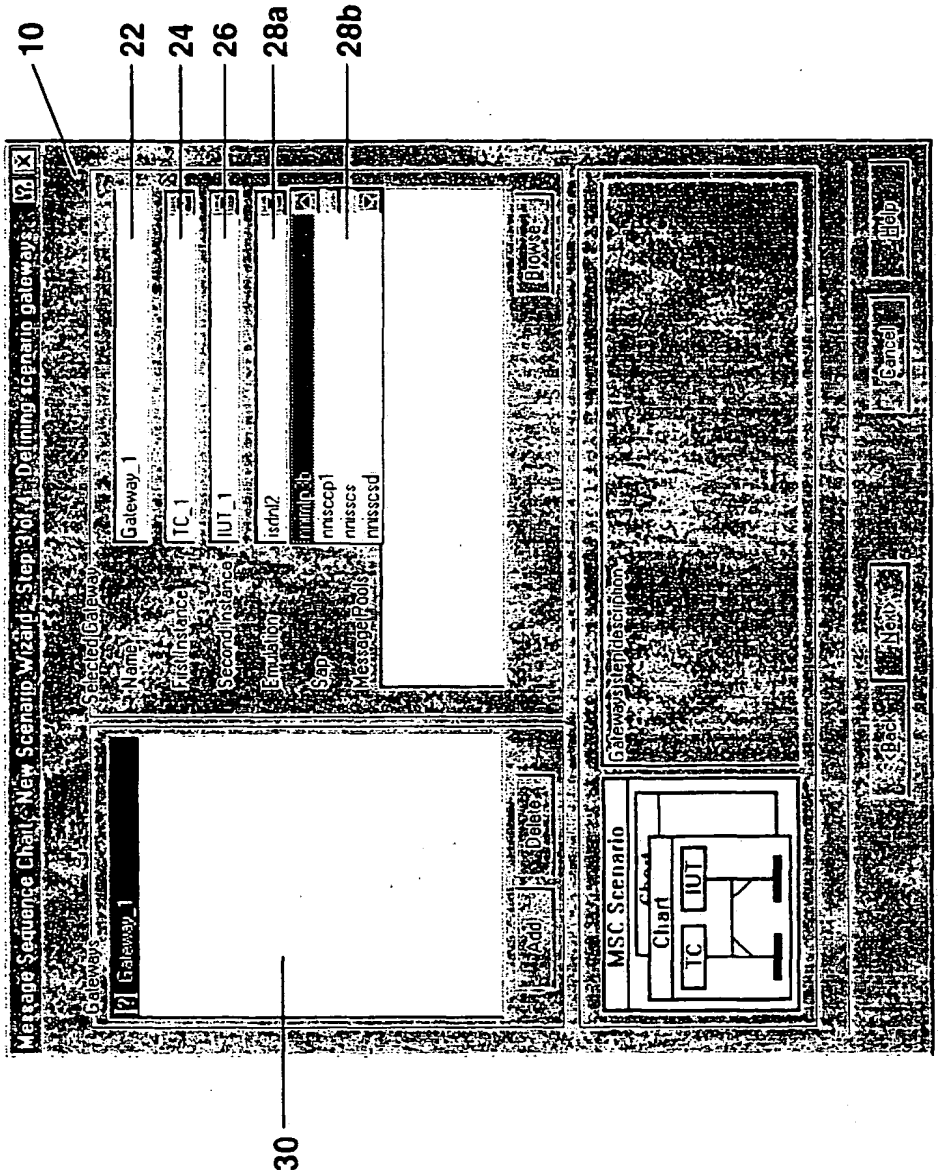


Fig. 2

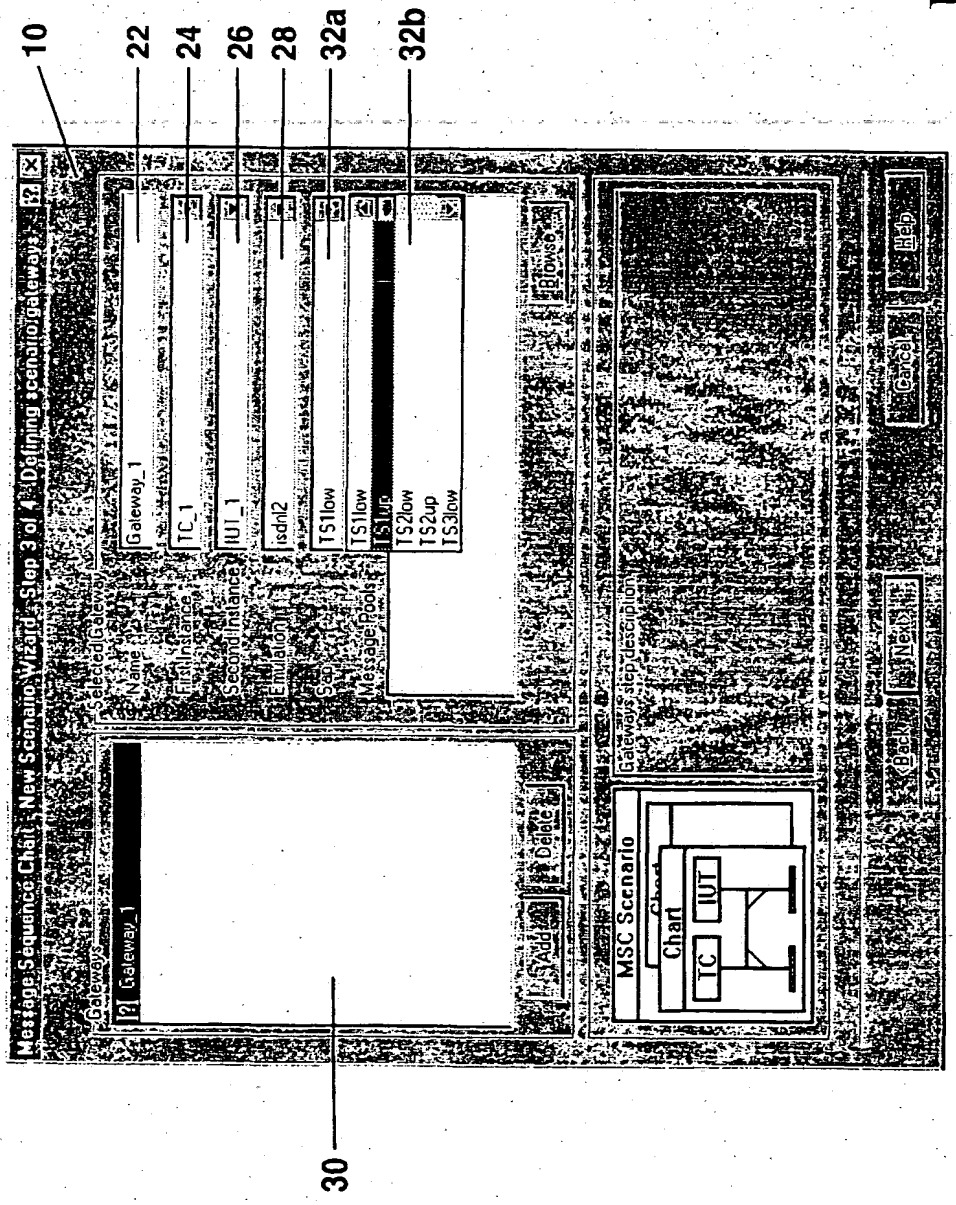


Fig. 3

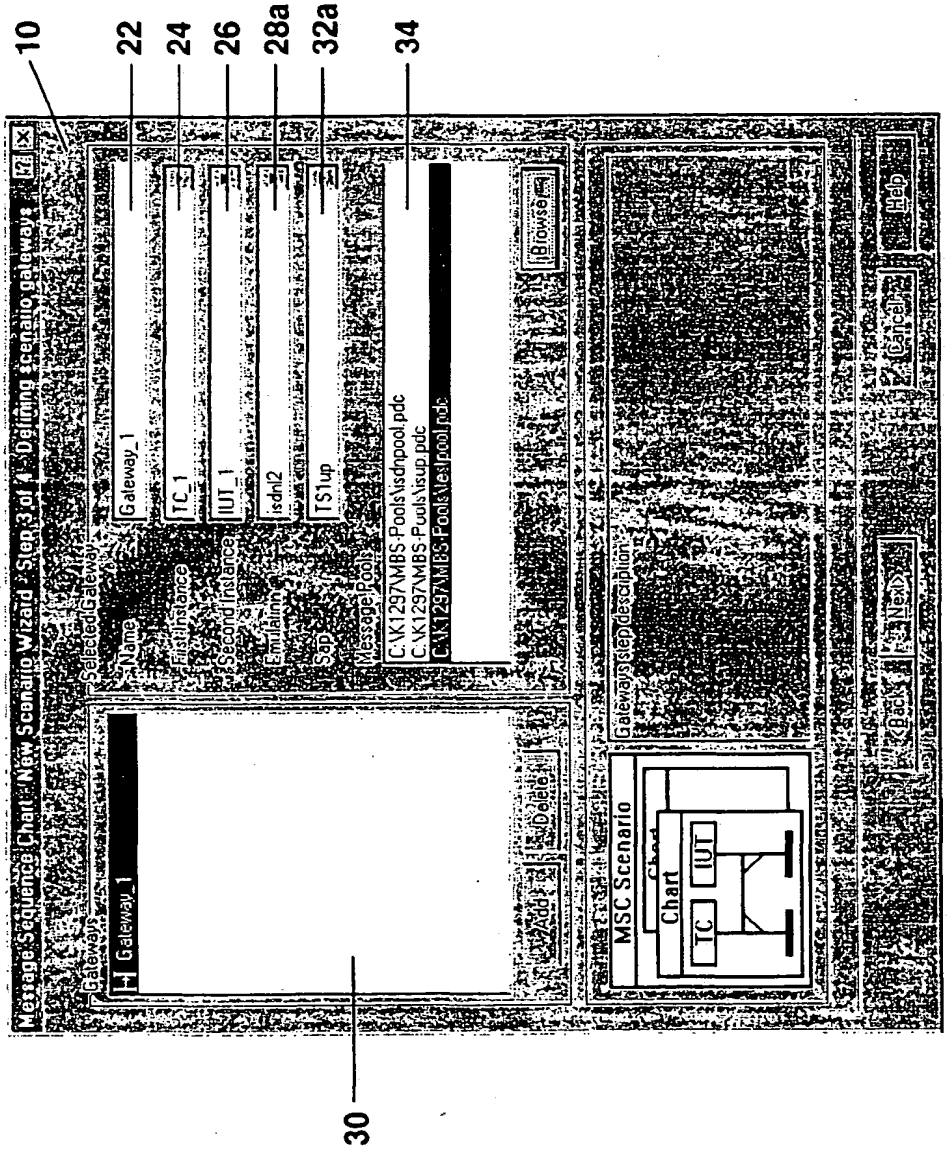


Fig. 4

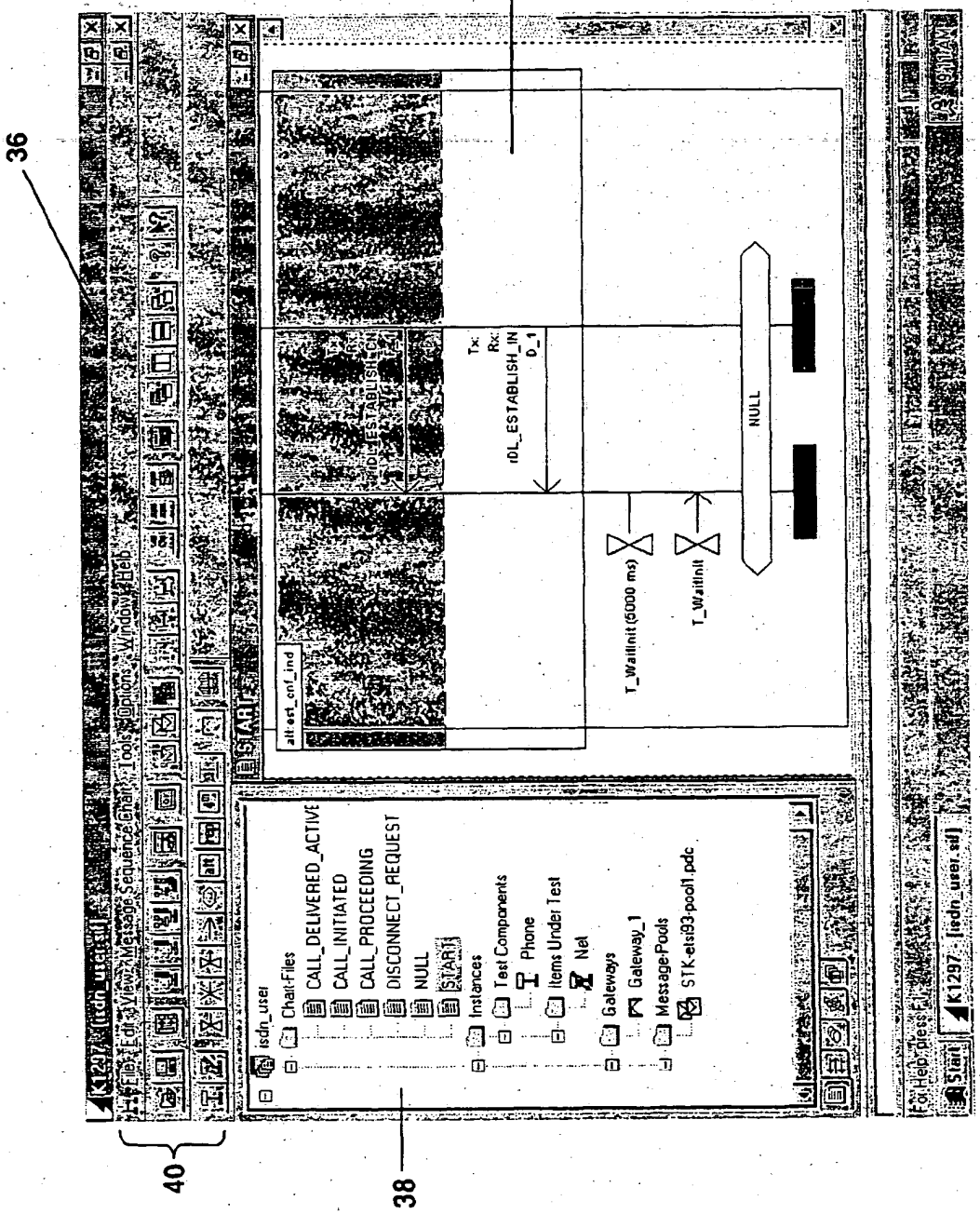


Fig. 5

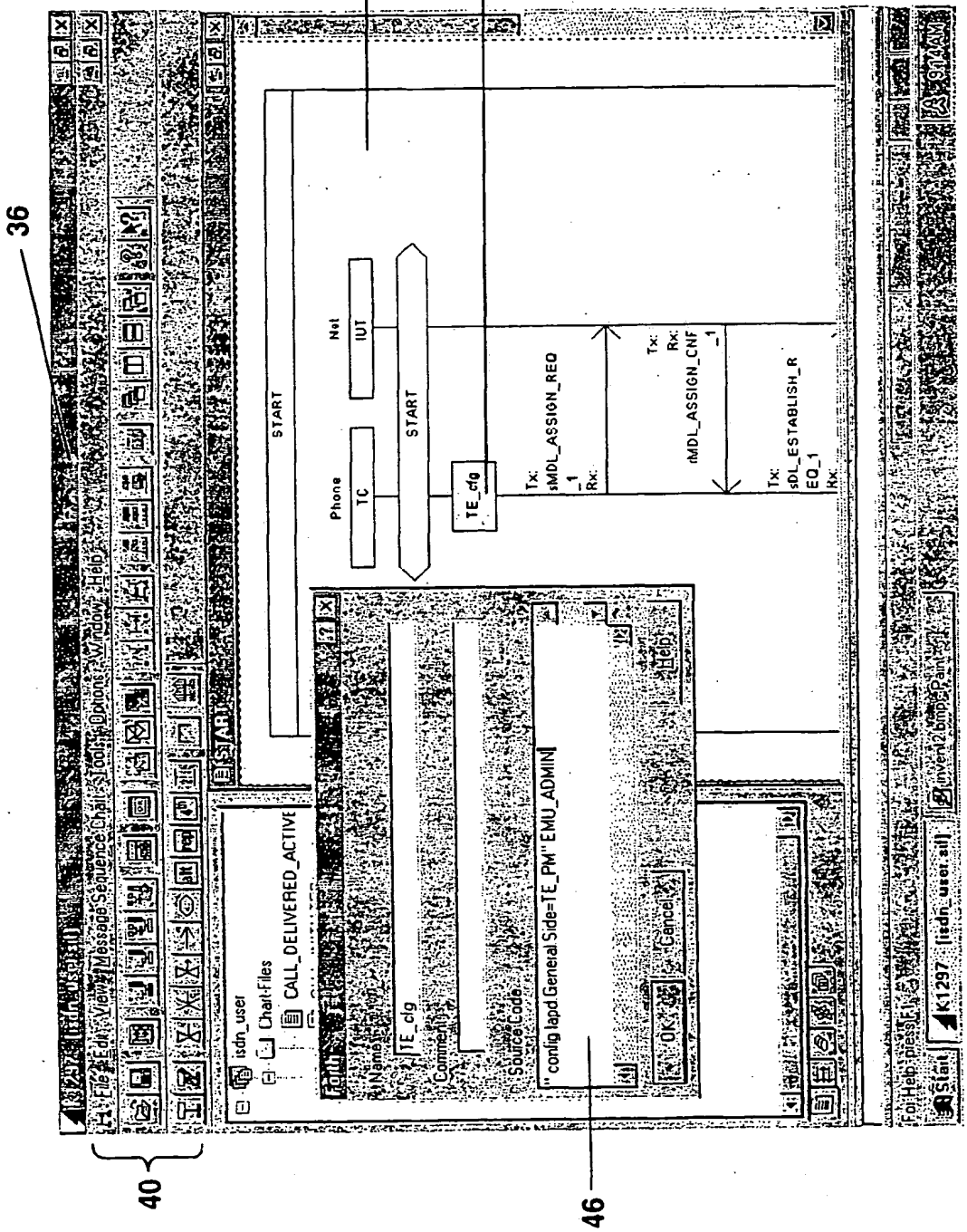

36
42
44
46
40
START
Phone
TC
Net
IUT
START
TE_clg
Tx:
sMDL_ASSIGN_REQ
_1
Rx:
Tx:
Rx:
rMDL_ASSIGN_CNF
_1
Tx:
sDL_ESTABLISH_R
EQ_1
Rx:
isdn_user
Chart-Files
CALL_DELIVERED_ACTIVE
TE_clg
Comment:
SourceCode:
"config lapd General Side=TE_PM" EMU_ADMIN
OK
Cancel
Help
Start
isdn_user.all

Fig. 6

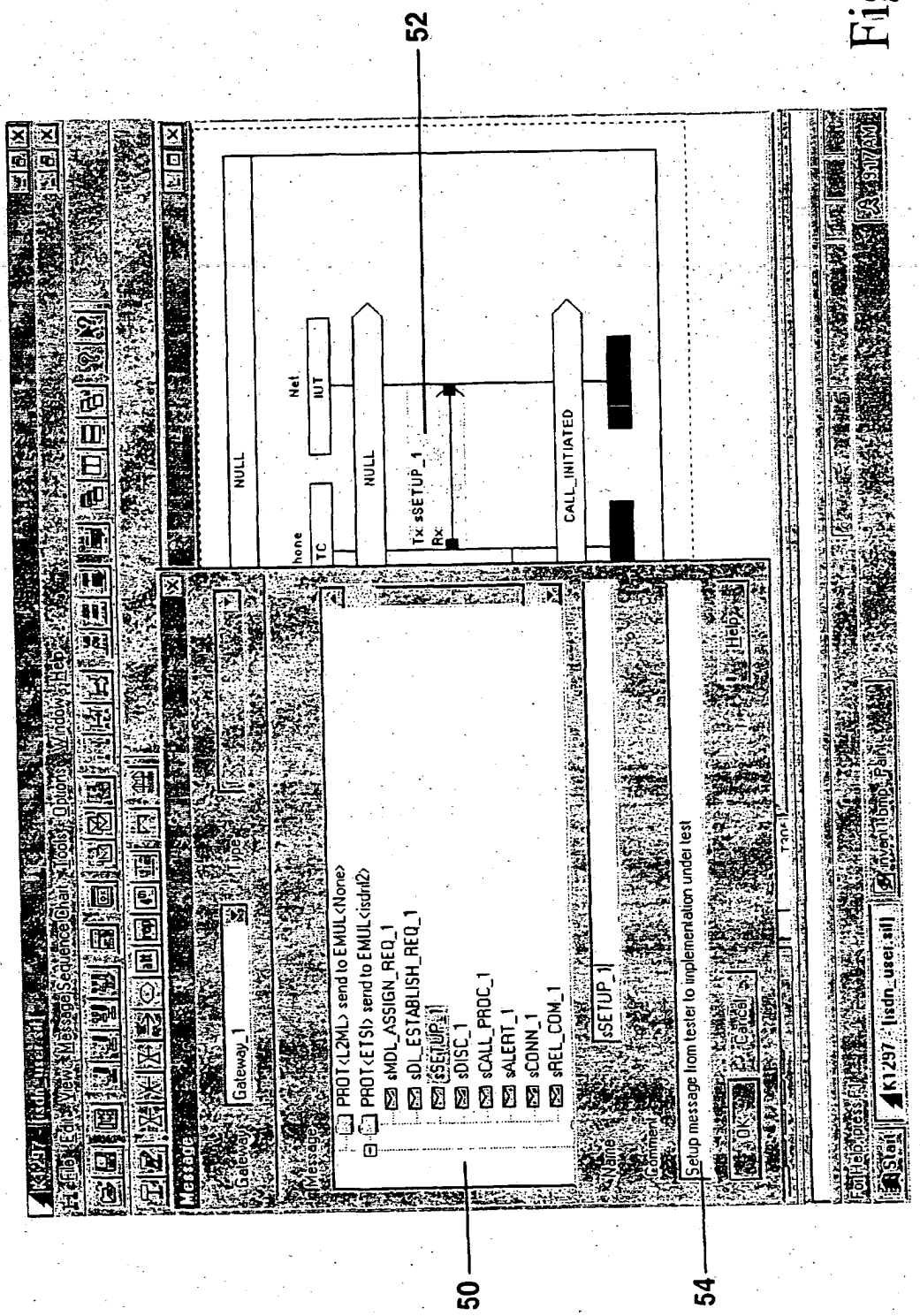


Fig. 7

Europäisches Patentamt

EUROPÄISCHER RECHERCHENBERICHT

Nummer der Anmeldung
EP 00 10 3798

EINSCHLÄGIGE DOKUMENTE

Kategorie	Kennzeichnung des Dokuments mit Angabe, soweit erforderlich, der maßgeblichen Teile	Betrifft Anspruch	KLASSIFIKATION DER ANMELDUNG (Int.Cl.7)
A	MULLER N J: "DESIGN AND CONQUER" BYTE,US,MCGRAW-HILL INC. ST PETERBOROUGH, Bd. 21, Nr. 10, 1. Oktober 1996 (1996-10-01), Seiten 93-94,96,98, XP000683573 ISSN: 0360-5280 * Seite 93, rechte Spalte, Zeile 12 - Seite 94, linke Spalte, Zeile 13 * * Seite 94, mittlere Spalte, Zeile 1 - Zeile 12 * ---	1-13	H04L12/24 H04L29/06 H04Q11/04
A	US 5 715 432 A (EPSTEIN MICHAEL L ET AL) 3. Februar 1998 (1998-02-03) * Spalte 1, Zeile 50 - Spalte 2, Zeile 41 * ---	1-13	
A	WO 98 57268 A (MCI COMMUNICATIONS CORP) 17. Dezember 1998 (1998-12-17) * Seite 5, Zeile 2 - Zeile 11 * * Abbildungen 2,4A * ---	1-13	
A	NEUENDORFF H: "UEBERTRAGUNGSQUALITAET IM MODERNEN KOMMUNIKATIONSNETZ STANDARDISIERT PRUEFEN UND SICHERN TTCN-COMPILER ERZEUGT AUF PROTOKOLTESTER K1197 GENORMTE KONFORMITAETSTESTS NACH ISO 9646-3 FUER ISDN UND CCS7" TELCOM REPORT,DE,SIEMENS AG. MUNCHEN, Bd. 17, Nr. 2, 1. März 1994 (1994-03-01), Seiten 76-77, XP000441333 ISSN: 0344-4724 * Seite 76 - Seite 77 * -----	1-13	RECHERCHIERTE SACHGEBIETE (Int.Cl.7) H04L G06F H04Q

Der vorliegende Recherchenbericht wurde für alle Patentansprüche erstellt

Recherchenort	Abschlußdatum der Recherche	Prüfer
BERLIN	6. September 2000	Siebel, C

KATEGORIE DER GENANNTEN DOKUMENTE

X : von besonderer Bedeutung allein betrachtet
Y : von besonderer Bedeutung in Verbindung mit einer anderen Veröffentlichung derselben Kategorie
A : technologischer Hintergrund
O : nichtschriftliche Offenbarung
P : Zwischenliteratur

T : der Erfindung zugrunde liegende Theorien oder Grundsätze
E : älteres Patentdokument, das jedoch erst am oder nach dem Anmeldedatum veröffentlicht worden ist
D : in der Anmeldung angeführtes Dokument
L : aus anderen Gründen angeführtes Dokument

& : Mitglied der gleichen Patentfamilie, übereinstimmendes Dokument

EPO FORM 1503 03.82 (P04C03)

ANHANG ZUM EUROPÄISCHEN RECHERCHENBERICHT ÜBER DIE EUROPÄISCHE PATENTANMELDUNG NR.

EP 00 10 3798

In diesem Anhang sind die Mitglieder der Patentfamilien der im obengenannten europäischen Recherchenbericht angeführten Patentdokumente angegeben.
Die Angaben über die Familienmitglieder entsprechen dem Stand der Datei des Europäischen Patentamts am
Diese Angaben dienen nur zur Unterrichtung und erfolgen ohne Gewähr.

06-09-2000

Im Recherchenbericht angeführtes Patentdokument	Datum der Veröffentlichung	Mitglied(er) der Patentfamilie	Datum der Veröffentlichung
US 5715432 A	03-02-1998	KEINE	
WO 9857268 A	17-12-1998	AU 7825498 A	30-12-1998

EPO FORM P0461

Für nähere Einzelheiten zu diesem Anhang : siehe Amtsblatt des Europäischen Patentamts, Nr.12/82

US 20010015732A1

(19) **United States**

(12) **Patent Application Publication** (10) **Pub. No.: US 2001/0015732 A1**

Ehrhardt et al. (43) **Pub. Date: Aug. 23, 2001**

(54) **SETTING UP A COMMUNICATION PROCEDURE BETWEEN INSTANCES AND A PROTOCOL TESTER USING THE METHOD**

(76) Inventors: **Joerg Ehrhardt**, Berlin (DE); **Jens Kittan**, Schwante (DE); **Wolfgang Borgert**, Berlin (DE)

Correspondence Address:
Francis I. Gray, Esq.
Tektronix, Inc.
P.O. Box 500
(50-LAW)
Beaverton, OR 97077 (US)

(21) Appl. No.: **09/776,040**

(22) Filed: **Feb. 1, 2001**

(30) **Foreign Application Priority Data**

Feb. 23, 2000 (EP) ..... 00 103 798 .5

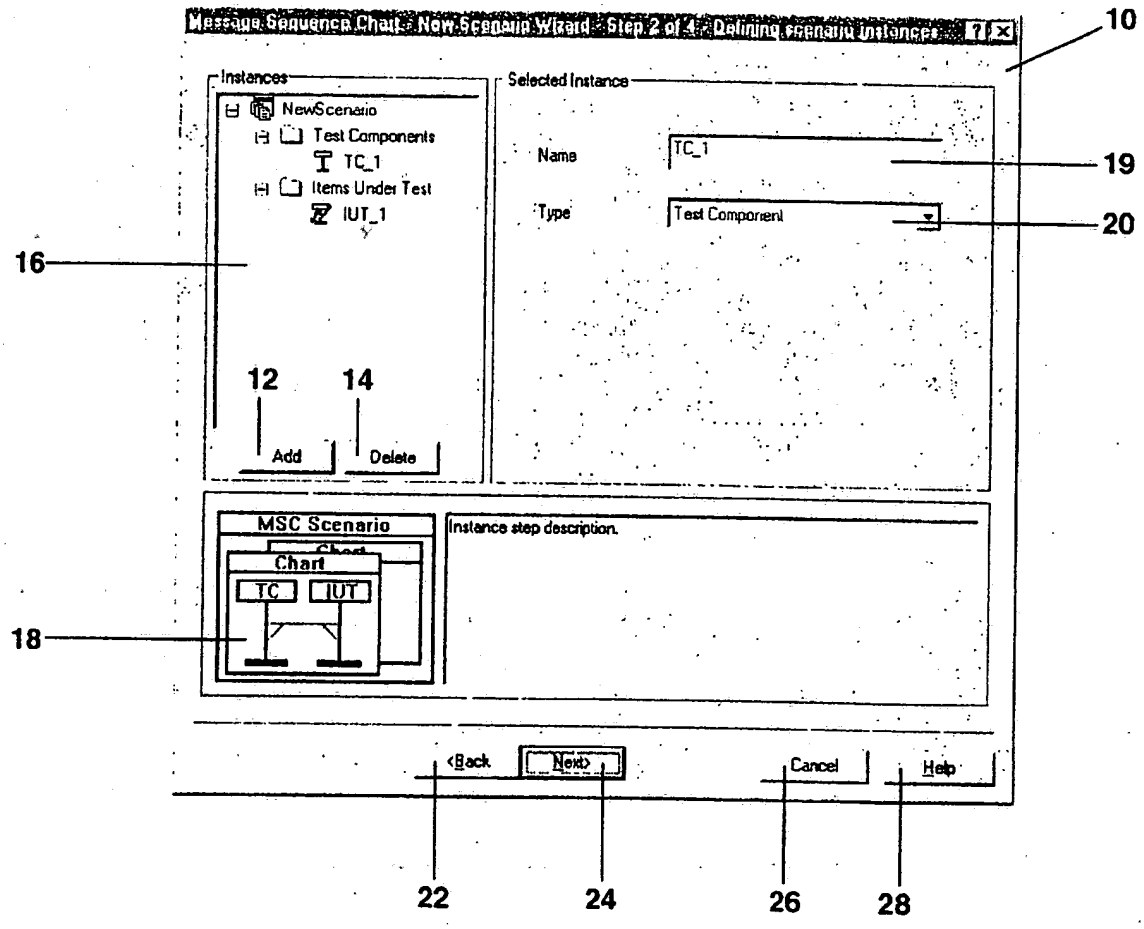
Publication Classification

(51) **Int. Cl.**[7] **G06F 15/00**

(52) **U.S. Cl.** **345/700**; 345/810

(57) **ABSTRACT**

A method of setting up a communication procedure between instances, one of which is a protocol tester that uses the method, includes the steps of selecting the instances, selecting a protocol layer for the communication procedure, selecting abstract communication interfaces for the protocol layer; selecting communication data and automatically setting up the communication procedure on the basis of the results of the selecting steps. Any of the selecting steps may be performed graphically, and parameters selected are assigned description files that are used in the setting up step.



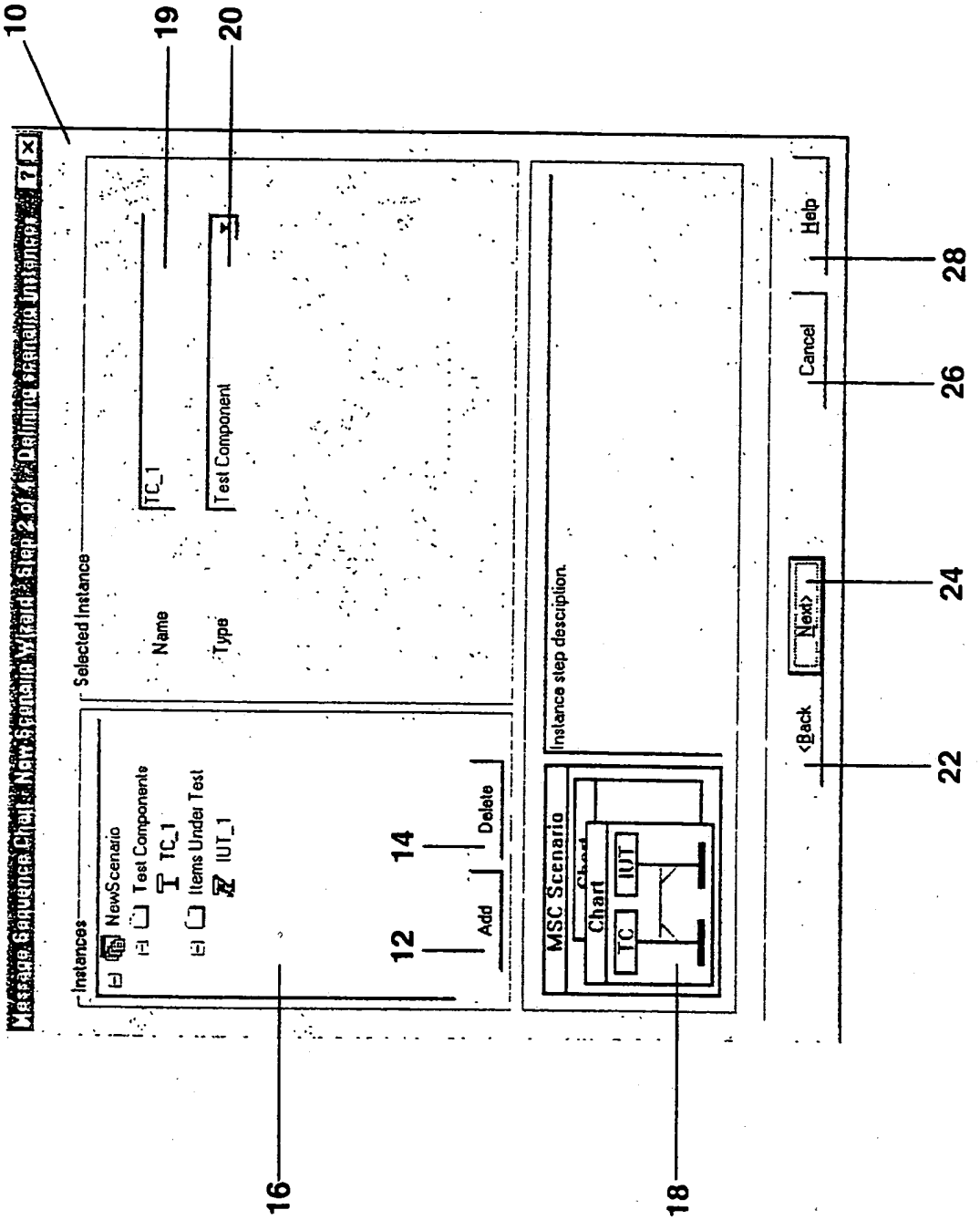

10
19
20
Message Sequence Chart: New Scenario Wizard: Step 2 of 4: Defining Scenario Instances
Instances
NewScenario
Test Components
TC_1
Items Under Test
IUT_1
Selected Instance
Name
TC_1
Type
Test Component
Add
Delete
12
14
16
MSC Scenario
Chart
TC
IUT
18
Instance step description
Back
Next>
Cancel
Help
22
24
26
28

Fig. 1

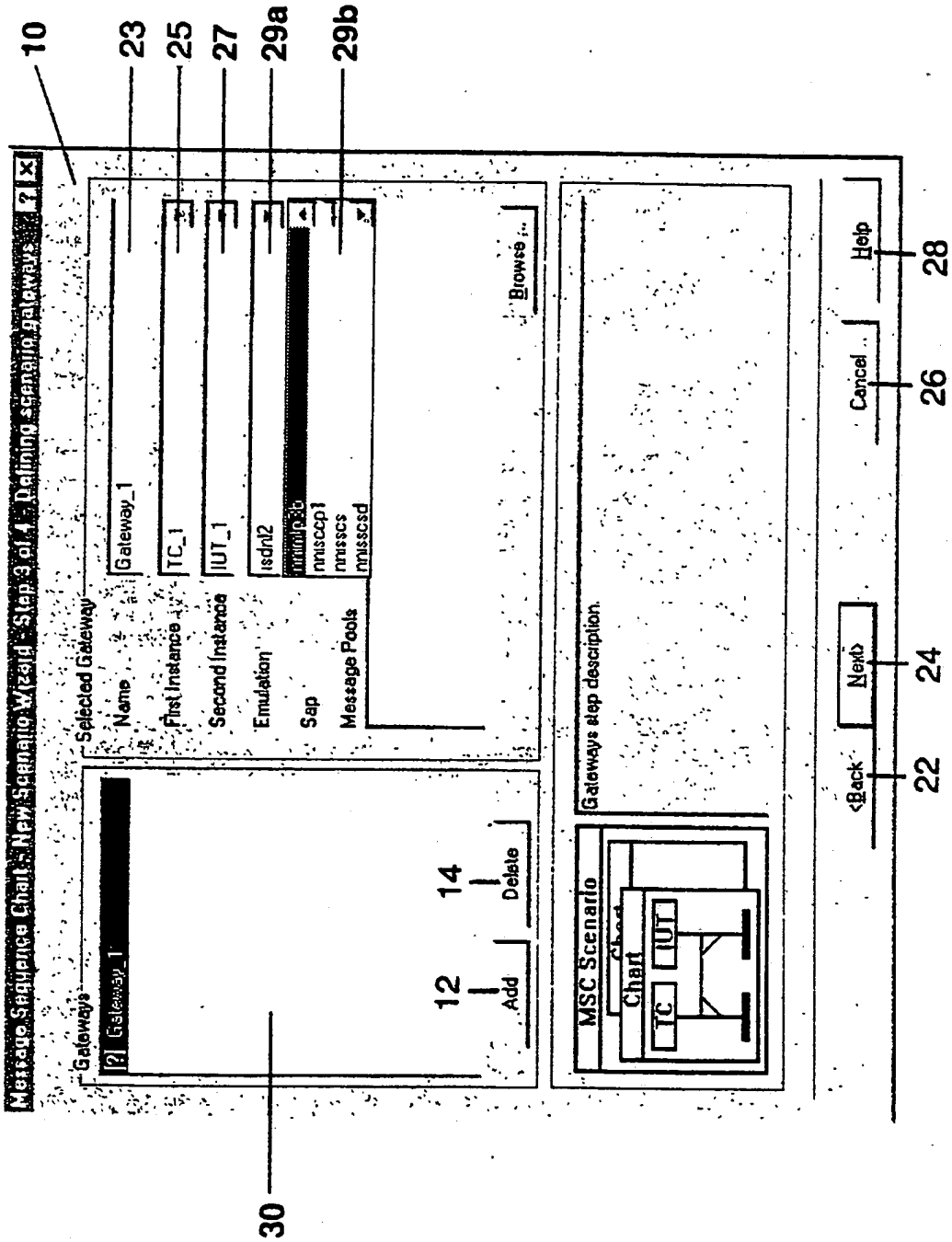

10
23
25
27
29a
29b
Message Sequence Chart: New Scenario Wizard - Step 3 of 4 - Defining scenario gateways
Selected Gateway
Name
Gateway_1
First Instance
TC_1
Second Instance
IUT_1
Emulation
isdn12
Sap
nmisccp1
Message Pools
nmisscs
nmisscsd
Browse...
Gateways
Gateway_1
Add
Delete
12
14
Gateways step description
MSC Scenario
Chart
TC
IUT
Help
Cancel
Next>
<Back
28
26
24
22
30

Fig. 2

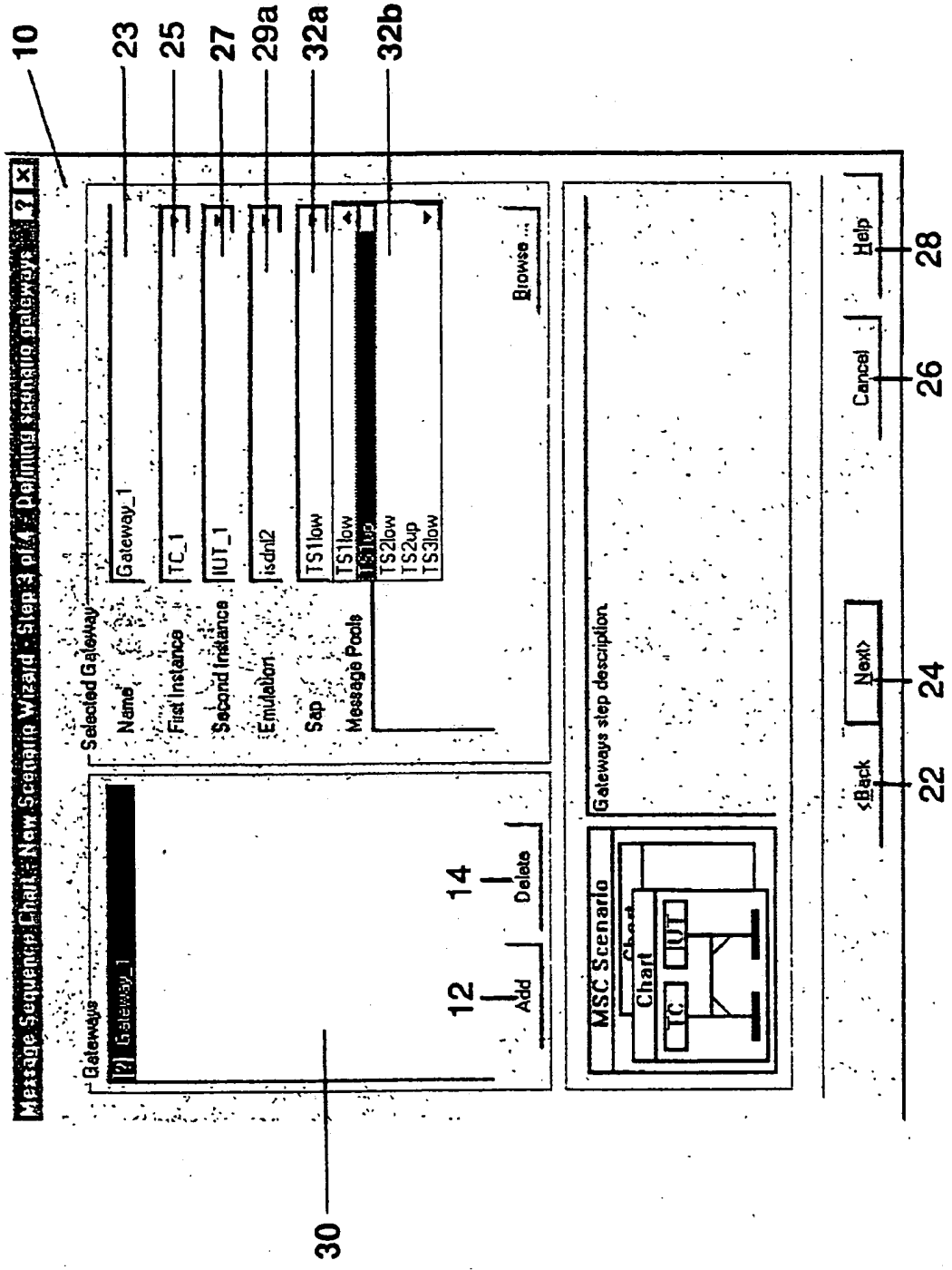

Message Sequence Chart - New Scenario Wizard - Step 3 of 4 - Defining scenario gateways
Gateways
Gateway_1
Add
Delete
Selected Gateway
Name
Gateway_1
First Instance
TC_1
Second Instance
IUT_1
Emulation
isdn2
Sap
TS1low
Message Pools
TS1low
TS1up
TS2low
TS2up
TS3low
Browse ...
Gateways step description
MSC Scenario
Chart
TC
IUT
<Back
Next>
Cancel
Help
10
12
14
22
23
24
25
26
27
28
29a
30
32a
32b

Fig. 3

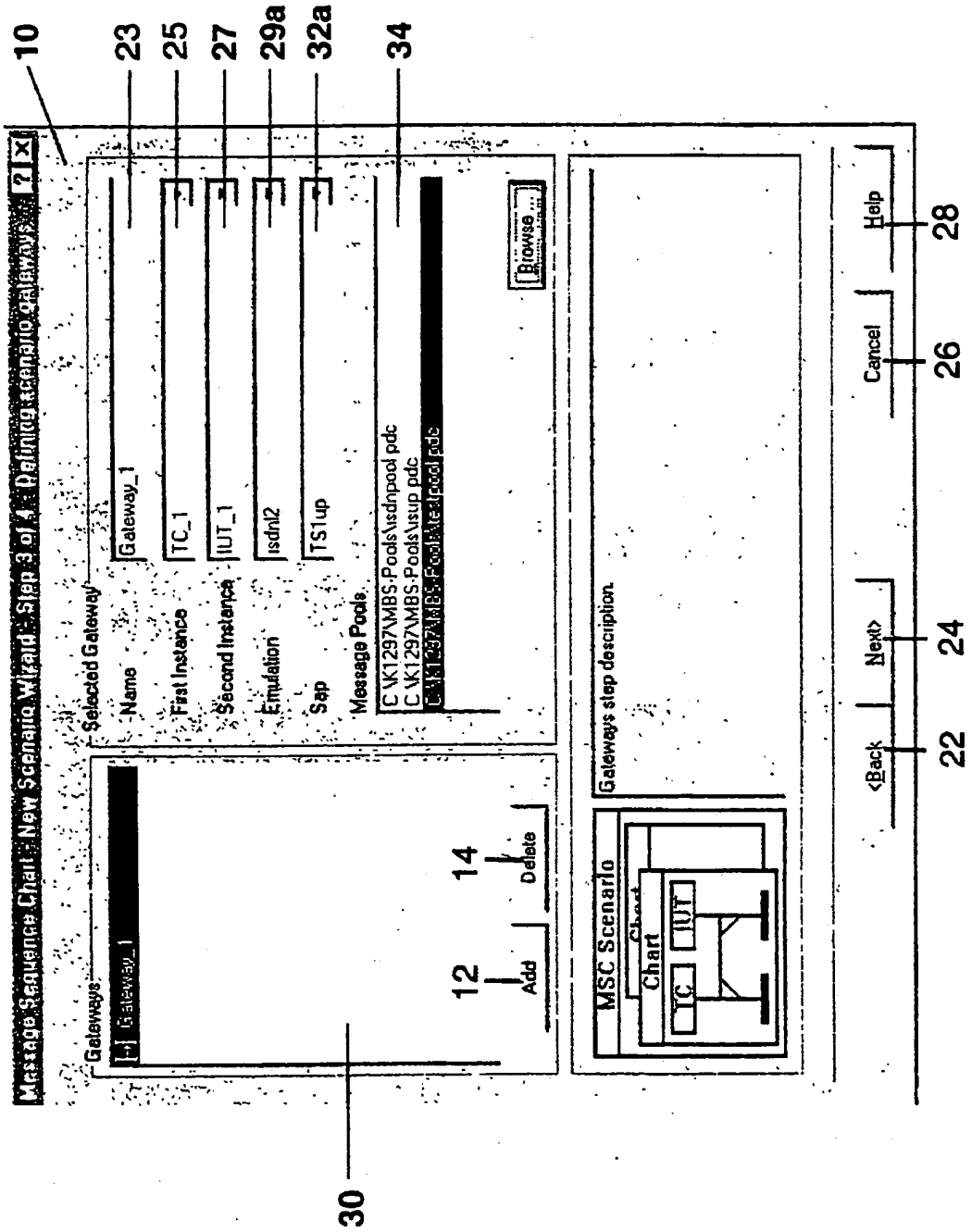

Message Sequence Chart - New Scenario Wizard - Step 3 of 4 - Defining scenario gateways
Gateways
Gateway_1
Add
Delete
Selected Gateway
Name
Gateway_1
First Instance
TC_1
Second Instance
IUT_1
Emulation
isdn12
Sap
TS1up
Message Pools
C:\K1297\MBS-Pools\isdnpool.pdc
C:\K1297\MBS-Pools\sup.pdc
Browse...
Gateways step description
MSC Scenario
Chart
TC
IUT
<Back
Next>
Cancel
Help
10
12
14
22
23
24
25
26
27
28
29a
30
32a
34

Fig. 4

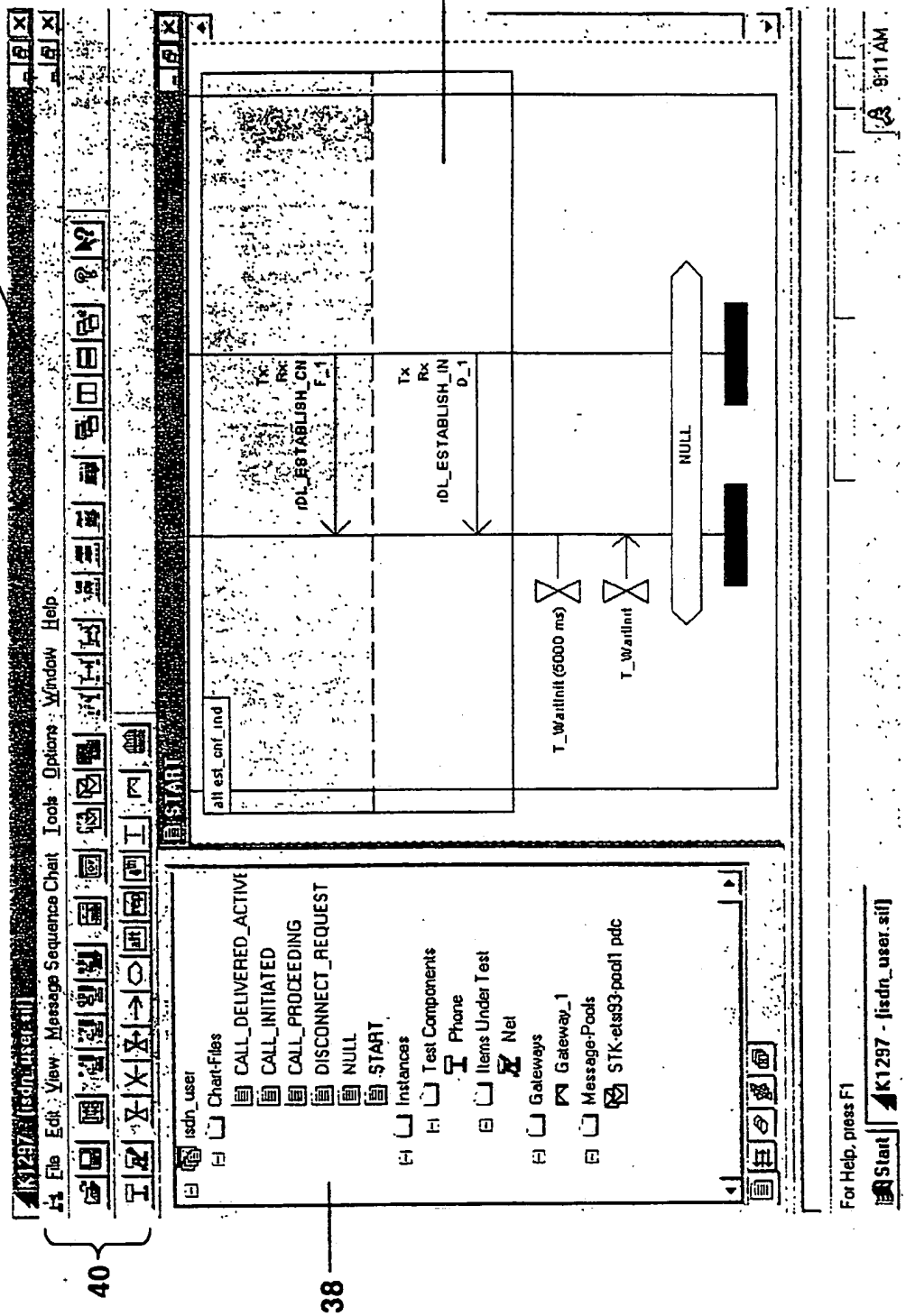

36
40
38
48
File Edit View Message Sequence Chart Tools Options Window Help
isdn_user
Chart-Files
CALL_DELIVERED_ACTIVE
CALL_INITIATED
CALL_PROCEEDING
DISCONNECT_REQUEST
NULL
START
Instances
Test Components
Phone
Items Under Test
Net
Gateways
Gateway_1
Message-Pools
STK-ets93-pool1 pdc
START
all est_cnf_ind
Tx
Rx
rDL_ESTABLISH_CNF_1
Tx
Rx
rDL_ESTABLISH_IND_1
T_WaitInit (5000 ms)
T_WaitInit
NULL
For Help, press F1
Start
K1297 - [isdn_user.sil]
9:11 AM

Fig. 5

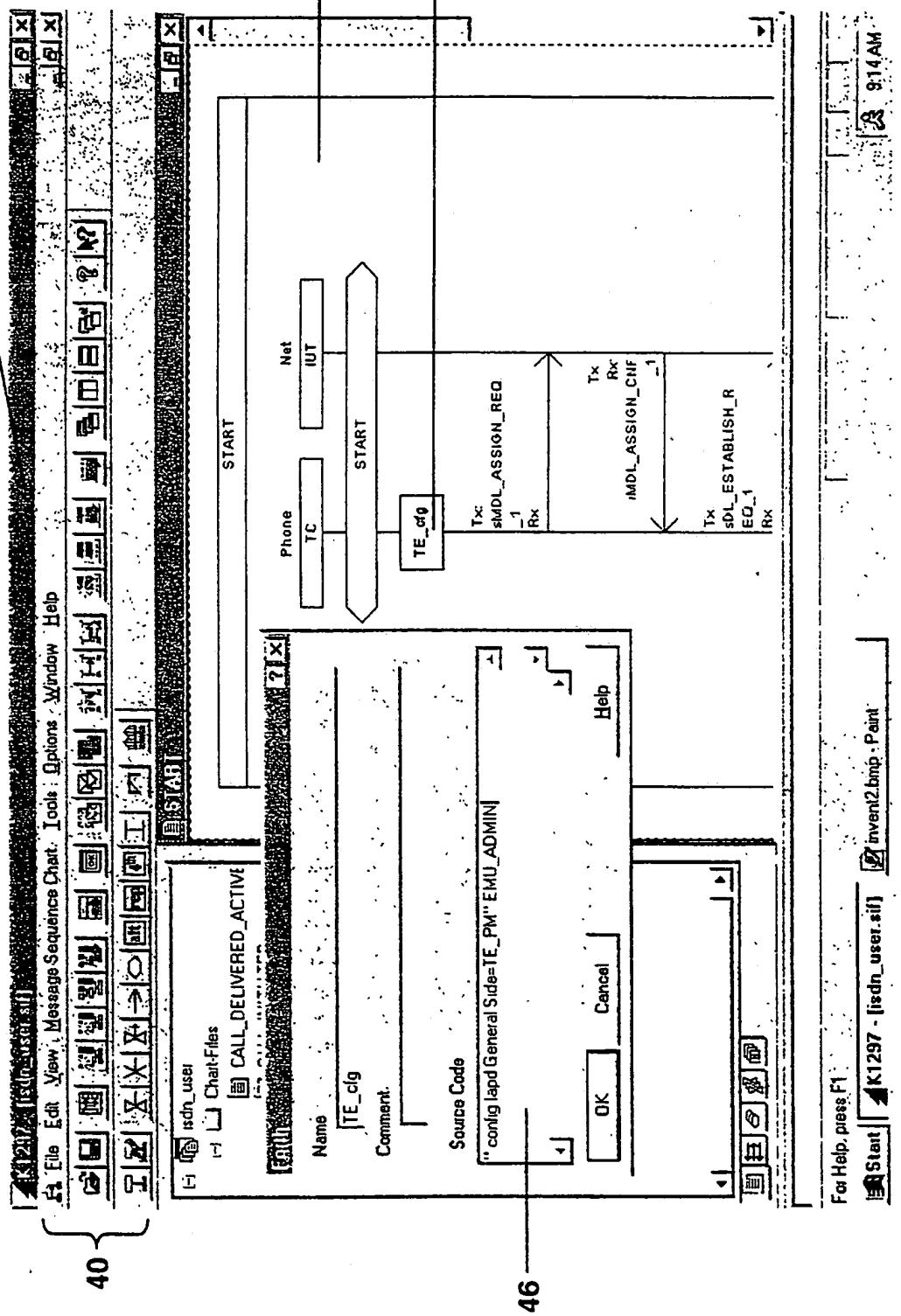

36
40
42
44
46
File Edit View Message Sequence Chart Tools Options Window Help
START
Phone
TC
Net
IUT
START
TE_clg
Tx
sMDL_ASSIGN_REQ
_1
Rx
Tx
Rx
rMDL_ASSIGN_CNF
_1
Tx
sDL_ESTABLISH_R
EQ_1
Rx
isdn_user
Chart-Files
CALL_DELIVERED_ACTIVE
Name
TE_clg
Comment
Source Code
" config lapd General Side=TE_PM" EMU_ADMIN
OK
Cancel
Help
For Help, press F1
Start
K1297 - [isdn_user.sil]
invent2.bmp - Paint
9:14 AM

Fig. 6

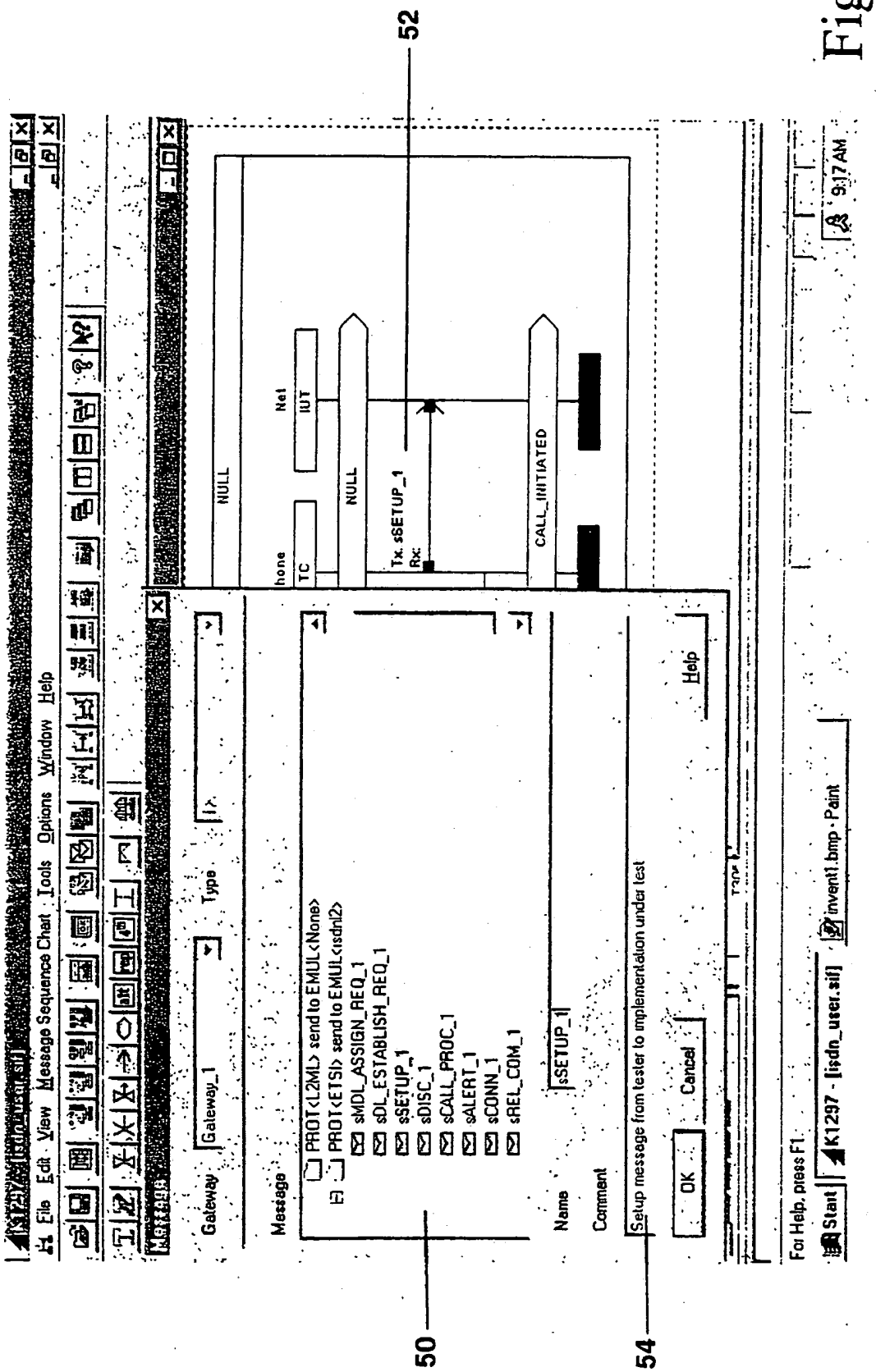

52
50
54
File Edit View Message Sequence Chart Tools Options Window Help
Gateway_1
Type
Message
PROT<L2ML> send to EMUL<None>
PROT<ETSI> send to EMUL<isdn2>
sMDL_ASSIGN_REQ_1
sDL_ESTABLISH_REQ_1
sSETUP_1
sDISC_1
sCALL_PROC_1
sALERT_1
sCONN_1
sREL_COM_1
Name
sSETUP_1
Comment
Setup message from tester to implementation under test
OK
Cancel
Help
NULL
Net
IUT
TC
NULL
Tx: sSETUP_1
Rx:
CALL_INITIATED
For Help, press F1
Start
K1297 - [isdn_user.sil]
invent1.bmp - Paint
9:17 AM

Fig. 7

SETTING UP A COMMUNICATION PROCEDURE BETWEEN INSTANCES AND A PROTOCOL TESTER USING THE METHOD

BACKGROUND OF THE INVENTION

[0001] The present invention relates to protocol testing, and more particularly to a method of setting up a communication procedure between instances, with one of the instances being a protocol tester, and to a protocol tester using the method.

[0002] In the field of protocol testing it is necessary to clearly specify a communication procedure by which a test is described so that the procedure may be executed automatically by a protocol tester. Languages such as TTCN (Tree and Tabular Combined Notation) make this possible, but they are complex and difficult to understand for an untrained reader. TTCN has prevailed in the field of Conformance Testing because these tests are very comprehensive, and TTCN supports such comprehensive tests very well. Apart from that, there are various proprietary test description languages. To facilitate understanding a standardized language, MSC (Message Sequence Charts), is used for the purpose of documenting and describing simple procedures. Further details on MSC may be taken from ITU-T Z.120, the contents of which are incorporated herein by reference. MSC consists of standardized process flow diagrams, also referred to as arrow diagrams or X diagrams. These diagrams may be understood independent of programming language. However, automatic execution of communications described by MSC is not possible on protocol testers. To obtain tests that are executable it is, therefore, necessary to write so-called "scripts", which requires that the user becomes thoroughly acquainted with the relevant programming language. In addition it is necessary to prepare documentation that is generally understood. For a test it is, therefore, necessary on the one hand to prepare graphical and textual documentation and on the other hand a source code or binary code that may be executed.

[0003] This state of the art results in a number of disadvantages. It is frequently necessary to convert existing tests, so there is a risk of inconsistencies. The test communication specifications often do not contain information on the configuration, or at least not in a format that may be read by a machine or by man. The different languages often represent proprietary approaches, which differ from equipment to equipment and have to be learned anew. The user is not supported or only receives rudimentary support with protocol knowledge when creating the messages and events.

[0004] What is desired is a method, and protocol tester using the method, that overcomes the above-noted disadvantages.

BRIEF SUMMARY OF THE INVENTION

[0005] Accordingly the present invention provides a method of setting up a communication procedure between instances, one of which is a protocol tester, by executing the following steps on the protocol tester: selecting instances that are to take part in the communication procedure; selecting a protocol layer on the basis of the communication procedure; selecting abstract communication interfaces of the protocol layer for the communication procedure; selecting communication data; and automatically setting up through the protocol tester on the basis of the above selections the communication procedure. The selections at any one of the steps may be made graphically with parameters selected being assigned description files that are used in the setting up step.

[0006] The objects, advantages and other novel features of the present invention are apparent from the following detailed description when read in conjunction with the appended claims and attached drawing.

BRIEF DESCRIPTION OF THE SEVERAL VIEWS OF THE DRAWING

[0007] **FIG. 1** is a plan view of a first graphical user interface (GUI) for a method according to the present invention.

[0008] **FIG. 2** is a plan view of a second GUI for a method according to the present invention.

[0009] **FIG. 3** is a plan view of the second GUI of **FIG. 2** in another presentation mode.

[0010] **FIG. 4** is a plan view of the second GUI of **FIG. 2** in a further presentation mode.

[0011] **FIG. 5** is a plan view of a third GUI for a method according to the present invention.

[0012] **FIG. 6** is a plan view of the third GUI of **FIG. 5** in another presentation mode.

[0013] **FIG. 7** is a plan view of the third GUI of **FIG. 5** in a further presentation mode.

DETAILED DESCRIPTION OF THE INVENTION

[0014] **FIG. 1** shows a graphical user interface (GUI) **10** that allows in a first step graphically selecting instances taking part in a communication procedure. Graphical selection in connection means that a symbol or a text proposal is shown graphically on the GUI, such as on a personal computer (PC) screen, and may be selected by simple activation, i.e., by clicking on it with a "mouse." One of the instances is a protocol tester on which the method as described herein is made available, with the protocol tester in the present case emulating a component, TC_1. Using two buttons, "Add"**12** and "Delete"**14**, a user may add further instances or delete instances listed. In a field **16** the compilation of instances is listed, while in another field **18** the compilation is shown as a diagram. In another field **19** the name of the instance may be selected, and in a further field **20** the instance type is shown. Two buttons, "Back"**22** and "Next"**24**, allow the user to move from one level of the definition of the communication procedure to the next, both in the direction of more detailed specifications and in the direction of higher-level presentations. A "Cancel" button **26** allows leaving a level, meaning that the changes made are reset. A "Help" button **28** offers the user further support.

[0015] According to **FIG. 2**, which shows another representation of the GUI **10**, the present communication procedure has the name Gateway_1, as shown in field **23**. Taking part in the procedure is a first instance TC_1, according to field **25**, and a second instance IUT_1, according to field **27**. According to field **29*a*** the emulated protocol is of the type "isdn12", with field **29*b*** offering further protocols from

which to choose. In field **30** various communication procedures that may be chosen for further processing may be offered. Buttons **12, 14, 22, 24, 26, 28** described with reference to **FIG. 1** appear again in similar form and with similar functions.

[0016] **FIG. 3** shows the GUI of **FIG. 2** in a different presentation mode, in this case for selecting a Service Access Point (SAP), as shown in field **32***a*. In field **32***b* there are further SAPs from which to choose. All SAPs shown in field **32***b* are offered for the selected emulation "isdn12."

[0017] **FIG. 4** shows another presentation form of the GUI **10** of **FIG. 2**, with a format for the communication data (Abstract Service Primitives—ASPs, Protocol Data Units—PDUs) now being used in a field **34** having so-called Message Pools.

[0018] **FIG. 5** shows another GUI **36** that provides the user with various types of information in a field **38**. First the instances selected by the user, then the test scenario (Gateway_1) agreed in accordance with FIGS. **1-4**, and finally the data format (Message Pools).

[0019] The following initially only refers to **FIG. 6**. The GUI **36** shows a large number of buttons **40** that, as is known from word processing or graphics programs, may be clicked by using a mouse. Using these buttons **40** the user may graphically set up the communication procedure in field **42**. **FIG. 6** shows the possibility of incorporating codes in the programming language Forth (Draft Proposal ANSI Standard 1994) into a block TE_cfg 44 by using an entry mask **46**. To enter codes in another programming language other entry masks may be used.

[0020] Returning to **FIG. 5** as an example of a part of the communication procedure shown in field **48**, an instance is awaiting, initially alternatively, the ASPs DL_ESTABLISH-_CNF_1 or DL_ESTABLISH_IND_1. Next a timer T_Waitinit with a five (5) second duratioin is started and the elapsing of the time is awaited.

[0021] **FIG. 7** shows an isdn-PDU "SETUP_1" being incorporated into a flow diagram prepared graphically as a send message. ASPs with PDUs from the Message Pool selected earlier are offered in an entry mask **50**. The PDU selected may be entered into a visually highlighted field **52**. In a field **54** the user is offered further information on the ASP or PDU selected.

[0022] In this way it is possible to set up the communication procedure, with preferably all selectable parameters being assigned description files that may be used with each other to automatically set up through the protocol tester the communication procedure to be executed between the instances.

[0023] When a code that may be executed is created, there are three interacting components. First the GUI stores the selected parameters, in particular the communication sequence, in an internal structure. Then a compiler translates the selected parameters into temporary files. Finally a linker reads the temporary files and converts them into the selected interpreter script language, such as ANS Forth. During this process the communication procedure as defined by the user is written into a script file.

[0024] Annex A1 shows the code automatically generated by the protocol tester for the figures described.

What is claimed is:

1. A method of setting up a communication procedure between instances, with one instance being a protocol tester, comprising the steps of:

selecting the instances that take part in the communication procedure;

selecting a protocol layer on the basis of the communication procedure;

selecting abstract communication interfaces of the protocol layer for the communication procedure;

selecting communication data; and

automatically setting up through the protocol tester the communication procedure on the basis of the selections made in the above selecting steps, with the abstract communication interfaces selecting and/or the communication data selecting steps being made graphically and parameters selectable during these steps being assigned description files that are used in the setting up step to set up the communication procedure.

2. The method as recited in claim 1 wherein the instances selecting step comprises the step of selecting the instances graphically, and/or the protocol layer selecting step comprises the step of selecting the protocol layer graphically, and the parameters selectable in these steps being assigned description files that are used in the setting up step.

3. The method as recited in claims **1** or **2** wherein the abstract communication interfaces comprise Service Access Points (SAPs).

4. The method as recited in claim 3 wherein the communication data comprise at least one type selected from the group consisting of Protocol Data Units (PDUs) and Abstract Service Primitives (ASPs).

5. The method as recited in claims **1** or **2** wherein the communication data comprise at least one type selected from the group consisting of Protocol Data Units (PDUS) and Abstract Service Primitives (ASPs).

6. The method as recited in claim 1 wherein the communication data selecting step comprises the steps of:

graphically selecting a data format; and

graphically setting up a communication sequence between the selected instances.

7. The method as recited in claim 6 wherein the graphically setting up step comprises the step of entering source code.

8. A protocol tester comprising:

means for selecting instances taking part in a communication procedure, one of the instances being the protocol tester;

means for selecting a protocol layer on the basis of the communication procedure;

means for selecting abstract communication interfaces of the protocol layer for the communication procedure;

means for selecting communication data; and

means for automatically setting up the communication procedure through the protocol tester on the basis of the selections of the various selecting means, with the abstract communication interfaces selecting means and/or the communication data selecting means being

graphical selecting means and parameters selected by the various selecting means being assigned description files that are used in the automatically setting up means for setting up the communication procedure.

9. The protocol tester as recited in claim 8 wherein the instances selecting means and/or the protocol layer selecting means comprise graphical selecting means and the parameters selected by these selecting means are assigned description files that are used in the automatically setting up means.

10. The protocol tester as recited in claims 8 or 9 wherein the abstract communication interfaces comprise Service Access Points (SAPs).

11. The protocol tester as recited in claim 10 wherein the communication data comprises one type selected from the group consisting of Protocol Data Units (PDUs) and Abstract Service Primitives (ASPs).

12. The protocol tester as recited in claim 11 further comprising means for entering source codes.

13. The protocol tester as recited in claim 8 wherein all parameters selected by all the selecting means are assigned description files that are used by the setting up means.

* * * * *